2003

Fractal compression and analysis on remotely sensed imagery

Ke Xiao
Louisiana State University and Agricultural and Mechanical College, kxiao@lsu.edu

FRACTAL COMPRESSION AND ANALYSIS ON REMOTELY SENSED IMAGERY

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Geography and Anthropology

by
Ke Xiao
B.E., University of Science and Technology, Harbin, China, 1982
M.S., Chinese Academy of Sciences, 1988
May 2003

ACKNOWLEDGEMENTS

This completion of the dissertation is the work based on many people's support and their support is gratefully, but not always adequately acknowledged. I will attempt to acknowledge some of my gratitude upon those that rallied round me for my dissertation study. Special thanks and immense appreciation and admiration goes with utmost respect to my major professor, Dr. Nina Lam. I cannot thank her enough for her selfless efforts and dedication to knowledge and her students. Her precious guidance, encouragement, patience, stimulation, and confidence both inspired and motivated me to obtain new frontiers of knowledge. Her support was felt throughout the entire program of my Ph.D. study. I was very fortunate to have professors as members of the final and general examination committee that were so knowledgeable and challenging of my abilities. Heartfelt and endless thanks are consequently expressed to the members of the final and general examination committee: Dr. Liu; Mr. Braud; Dr. Leitner; Dr. Huh; and Dr. Guzik. The committee provided boundless patience and assistance, including valuable comments and suggestions on my dissertation. Mr. Braud also benevolently provided the essential Landsat TM images for the study areas used in this research. Without these images, the scope of this dissertation research would be very limited. In addition, it is said that ordered minds open the door to knowledge. The memory, the ability to recognize objects, issues, and ideas and their applications to the universe of knowledge, and their ability to parse this information within the framework of their discipline is always invaluable. Consequently, I appreciate the efforts of my good friend, Mr. Dennis Watson, who reviewed the manuscript and made helpful and constructive comments.

All technical and scientific studies are affected greatly by the technical access and assistance available. In this study, I was provided a wealth of these resources. Subsequently, I would like to thank Mr. Jack Haynes and Mr. Farrell Jones, the former director and the system manager of the CADGIS Laboratory, for providing technical supports for my dissertation research in the lab. I also want to thank my fellow graduate students in the Department of Geography and Anthropology, the CADGIS lab as well as the CMS lab for their general succinct and stimulating intellectual discussions in research and specific technical issues and methods, particularly, Xiongping Zhang, Weiwen Feng, and Wei Zhao, together with our fantastic LSU faculty, staff, and the other students. They provided a pleasant and constructive atmosphere that immensely enhanced the surroundings for conducting my work in the department and the labs.

This research was partially supported by the benevolent Robert C. West fund from the Department of Geography and Anthropology, Louisiana State University.

Special thanks should be given to my parents, parents-in-law, and the whole family, in particular, my loving wife Xiaocheng Wu, and my devoted daughter Jessica, for their deep love, gracious understanding, sacrifice, and continual patience through the course of my whole study. This is what the true meaning of life is about, knowledge and relationships and certainly my loved ones are in my memory and on my mind. At one time, my studies took me to another city, a city that was remote as the moon because of time constraints and access. A short distance can be an eternity in relation to circumstances. My wife faithfully took the arduous responsibility of taking care of the family so that I could concentrate on my study and work. My family, including my wife, daughter, parents, and in-laws deserve this recognition.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Remote sensing images contain huge amount of geographical information and reflect the complexity of geographical features and spatial structures. As the means of observing and describing geographical phenomena, the rapid development of remote sensing has provided an enormous amount of geographical information. The massive information is very useful in a variety of applications but the sheer bulk of this information has increased beyond what can be analyzed and used efficiently and effectively. This uneven increase in the technologies of gathering and analyzing information has created difficulties in its storage, transfer, and processing. Fractal geometry provides a means of describing and analyzing the complexity of different geographical features in remotely sensed images. It also provides a more powerful tool to compress the remote sensing data than traditional methods. This study suggests, for the first time, the implementation of this usage of fractals to remotely sensed images.

In this study, based on fractal concepts, compression and decompression algorithms were developed and applied to Landsat TM images of eight study areas with different land cover types; the fidelity and efficiency of the algorithms and their relationship with the spatial complexity of the images were evaluated. Three research hypotheses were tested and the fractal compression was compared with two commonly used compression methods, JPEG and WinZip. The effects of spatial complexity and pixel resolution on the compression rate were also examined.

The results from this study show that the fractal compression method has higher compression rate than JPEG and WinZip. As expected, higher compression rates were obtained from images of lower complexity and from images of lower spatial resolution

(larger pixel size). This study shows that in addition to the fractal's use in measuring, describing, and simulating the roughness of landscapes in geography, fractal techniques were useful in remotely sensed image compression. Moreover, the compression technique can be seen as a new method of measuring the diverse landscapes and geographical features. As such, this study has introduced a new and advantageous passageway for fractal applications and their important applications in remote sensing.

CHAPTER 1

INTRODUCTION

Geographers and other geo-scientists have been exploiting an increasing number of techniques to observe and describe our planet; the ability to capture, store, and analyze this information about our planet has rapidly increased from primitive man and ancient civilizations to modern man. The primary method of acquiring data was manual surveying work in the field; the representation, the storage, and the analysis of geographic information was once in the form of different maps with text description. Today, coinciding with the development of science, technology, civilization, and especially computer science, space technology, electronic communication technologies, as well as increasing requirements for information, science has taken more advanced methods to observe the geographic phenomena, obtain geographic data, and record and present geographic information. We are increasingly understanding our universe and the Earth on which we live. There are more advanced tools to collect, store, retrieve, and analyze the collected data for a variety of applications. Since the 1970s, remote sensing (RS) has become one of the most useful tools that geo-scientists use to recognize earth system processes and to capture geographic information throughout global, regional, and local scales. We can collect and process multi-scale, multi-phase, multi-spectrum, multi-temporal, and multi-dimensional spatial data (Jensen 1996). Therefore this information collected by the use of modern technology has become an important data resource of Geographic Information System (GIS).

Remote sensing is a science, art, and technology; it is also a tool to recognize the world around us. Remote sensing is not a new subject, but rather a technique that has

grown explosively over the last decades in terms of investment, research, and the design of new techniques (Drury 1990). Since remote sensing was put into use, it has undergone multiple incremental advancements. The platforms have developed from balloon, to airplanes and satellites. The sensors have built-up from simple photosensitive imaging cameras using only visible light, to multiple-spectral scanners that sense signals non-discretely from ultraviolet to infrared spectra. Media has developed from photosensitive film to magnetic tape, to signals directly sent from the satellites. The applications have developed from single survey, regular landscape/land cover surveys to crop estimation, disaster prediction, weather broadcast, mineral exploration, city planning, environmental monitoring, and global change, etc. Remote sensing data have become increasingly essential, coincidentally with the development of remote sensing technologies together with sensors, digital picture processing, and computer hardware and software. Nowadays most natural resource mapping is done using remote sensing data. Aerial photography has been used to produce virtually all topographic maps and most forestry, geology, land use, and soils maps (Lillesand and Kiefer 1999). More recently, airborne radar and scanner data as well as satellite imagery are being used for mapping applications. Satellites have become the primary tool in the acquisition of global and synoptic detailed information about the planets (including the Earth) and their environments. Sensors on Earth-orbiting satellites provide information about global patterns and the dynamics of clouds, surface vegetation cover and its seasonal variations, surface morphological structures, ocean surface temperature, and near-surface wind. The rapid wide-coverage capability of satellite platforms allows the monitoring of rapidly changing phenomena, particularly in the atmosphere. The long duration and repetitive capability of the satellite allows the

observation of seasonal, annual, and more long-term changes such as polar ice cover, desert expansion, and tropical deforestation. The large-scale synoptic coverage of the satellite allows the observation and study of regional and continental scale features, including plate boundaries and mountain chains (Elach 1987).

The rapid development of remote sensing technology has provided the capability to generate data at a far greater rate than data can be analyzed and used. In fact, the development of sensor, data receiving, and data storage capabilities has proceeded faster than the development of practical applications. Often, those who could benefit most from the technology do not have the information or the context to assess this information. For this reason, remote sensing has been given a more prominent treatment than other data sources commonly used in geographic information systems (Star et al. 1997).

When remote sensing technologies are being developed, more and more categories of remotely sensed data and other spatial data are obtained to meet the various requirements of applications and analyses. The amount of information explodes: in many practical applications, people often encounter the digital images with 512 x 512 or an even larger number of pixels. For remotely sensed imagery, which is usually multi-band, the sizes are usually much larger. For example, a scene of MSS image has 2,340 (rows) x 3,240 (columns) x 7 (bands) pixels. Each pixel has 8 bits to represent 256 gray levels. So it needs 424,569,600 bits of storage; for the images taken from IKONOS, which was launched on September 1999, the storage needed are even bigger due to the high radiometric resolution: the IKONOS images use 11 bits to represent 2048 gray levels. On the other hand, the complexity and density of remote sensing data have also become much higher. For example, NASA Earth Observing satellites with global coverage such

as Landsat 7 and Terra can provide data at rates of about 1 terabyte per day. The range of spatial resolutions from these satellites varies from tens of meters to tens of kilometers, and with a wide range of wavelengths from visible to microwave, as well as a more frequently repeated coverage. Moreover, NASA EOS-1 hyperspectral remote sensing satellites will utilize hundreds of bands with resolution of tens of meters, producing hundreds of gigabytes of data sets for even small regional coverage. Evidently, the amount of information will be enormous.

1.1 Problem Statement

The recent and widespread introduction of geographic information systems has created a sudden need for users of GIS to become knowledgeable about this technology (Aronoff 1986). GIS technology has developed so rapidly over the past two decades that it is now accepted as an essential tool for the processing of geographic information. In the mean time, remote sensing is also becoming an important data source of geographic information systems, as remote sensing can provide high complexity and high-density accurate geographic information.

While the information consumers are using remote sensing to acquire both diversity and vast amounts of geographic data, especially the remotely sensed imagery, they find that they are faced with a problem: the vast quantities of data demand considerable data storage capacity and powerful means of data processing and data analyzing. Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demands for larger data storage capacity and data transmission bandwidth continue to outstrip the capabilities of available technologies (Saha 2002). On one hand, users require more and more remotely sensed images, which

provide large amount of geographic information. On the other hand, they have more and more difficulties to store, process, and transfer them. It is desirable to represent the remote sensing image information with fewer storage bits and when needed, to be able to reconstruct the original image so that it will facilitate transfer electronically as compressed images. It will also be necessary to again store the information with as little space as feasible and accurate. Image compression techniques can provide a way of solving this problem.

How does image compression work? It is applied to obtain an image representation while reducing the amount of memory needed as much as possible to encode the image (Gomes and Velho 1997). Image compression is possible because images, in general, are highly coherent (non-random), which means that there is repetitive information. Image data, like other meaningful data, are usually structured, and this structure means that the data over different parts of an image are interrelated. For example, consider an image in matrix format, if we take an arbitrary pixel, its gray-level or color will likely be of similar value to that of the neighboring pixels, since they are more likely than not to belong to the same object. If the gray levels or colors are not similar, some more complex relationship may apply: for instance, the pixel might be on the boundary of two objects, or it may be part of a texture pattern. In any case, there are usually some redundant or less informative data because of the image's structure. So image compression is basically a way to capture the image structure and generalize this image structure in coherent and usable form. Compression methods try to eliminate repetitiveness, thus producing a more compact code that preserves the essential and accurate information contained in the original image.

How can image compression solve the problem? Image compression is a thoroughly pragmatic operation. However, the efficiency of image compression depends on the image complexity and a number of geographical factors. Because images require large amounts of data, storing and transmitting this data places a significant load on the computer systems and data transmission facilities used. Compression of data reduces the cost of image storage by increasing the effectiveness of storage resources and increases the effective speed of transmission without "broad banding". So it is apparent and necessary to find and develop efficient compression and decompression algorithms, then putting them into practical use on remotely sensed imagery, and testing their effects on images reflecting different geographical phenomena. These are important problems in remote sensing and therefore, they are the main task of this research project.

1.2. Research Objectives

There have been many kinds of image compressing theories and techniques in the literature (Pratt 1978; Jain 1981; Eichmann et al. 1989; Gray et al. 1992; Hoffman and Johnson 1994; Lammi and Sarjakoske 1995; and Saha 2002). The basic technique is to use an invertible linear transformation to transform the given correlated image into an array of uncorrelated variables that can be represented with a smaller number of bits compared to the original image. Fourier transformation, cosine and sine transforms, Hadamard transform, and Karhunen-Loeve transform (KLT) are linear transforms, in which the KLT can produce the exact uncorrelated transform coefficients for finite images; however, the computational problems are complex. These problems include extended computing time and vast and expensive space resources; while other transforms, including Fourier, cosine, and sine, can be quickly carried out, but their coefficients are

not 'exactly' uncorrelated. Moreover, the compression rates in these transforms are limited, thus the results could potentially be inappropriate for the use. Along with the development of theories and techniques, some compression methods, including predictive encoding, are proposed to overcome the shortage of the previous ones, but the balance between high compression rate, less computational time and space, and high accuracy has still not been obtained.

'Fractal' is a new geometry proposed by French mathematician Mandelbrot in the 1970s (Mandelbrot 1977). It represents a revolution in topological space theory and provides the possibility of describing and simulating objects and phenomena accurately in the natural world by using the fractal geometry. The new mathematical model provides people with a new and different viewpoint to the natural world. Since it was advanced in the 1970's, the techniques based on fractals have also been applied in several areas of image processing, such as image segmentation, image analysis, computer graphics, and even in image compression (Barnsley 1988; Barnsley and Hurd 1993; Wohlberg and Jager 1999; Belloulata and Konrad 2002). However, there has not been an attempt to use the technology for remote sensing image compression. It is therefore appropriate that two research questions will be addressed in this study:

1) Can algorithms based on fractal theory be developed and implemented in remote sensing image compression?
2) Can fractal technique improve the compression rate and overcome the shortages of the previous image compression methods?

Based on the questions, the research objectives are twofold:

1) Develop algorithms to compress (encode) and decompress (decode) remotely

sensed imagery in order to compress and restore it by using fractal theory so that the contribution will be made in both theory and technique.

2) Test the fractal algorithms and examine how the performance of the method, including its compression rate, image fidelity, and computational time, varies with different geographical phenomena and landscapes in images.

Then the research hypotheses will be:

1) Fractal image compression method can be better than other commonly used ones.
2) The higher the images' spatial complexity, the lower the compression rate.
3) Changing scale or resolution of images will affect the compression rates

1.3 Expected Significance

The expected results and the significance include:

1) The development of the fractal compression and decompression algorithms provide a useful alternative to existing image compression methods.
2) A better understanding of the relationship between image complexity, computational complexity, and the effects of various types of geographical phenomena on the image compression technique.

In this study, both fractal image compressing and decompressing algorithms will be applied on images reflecting different landscapes that have different spatial complexities. Some images may be smooth, e.g., water bodies; some may be rough, as hybrid vegetation; and some may exhibit more structural information, as urban areas. The results, including computational time and compression rates, may be affected by geographic complexities. It is an important part of the study to analyze the effects of different landscapes on the

compression rate and image fidelity. The relationship between the landscapes and the image compression rates will be more clearly revealed, moreover, the effect of changing pixel resolution on image compression and how the scale affects change with image complexity can be understood.

3) Development in fractal technique

The fractal theory and technique will be developed for the compression of remotely sensed images. In the past, the fractal technique was used to measure, describe, and simulate the roughness of landscapes in geography, while in the area of image and graphic processing, it was used to generate some graphs and artificial images, including the moon surface, etc. This study will explore a new application in geography and remote sensing.

4) Advance in image processing

This study will also extend image compression theories and methods. By introducing the fractal technique, it is expected that higher compression rate and image fidelity can be obtained so that the storage as well as data handling problems for remote sensing images can be minimized. The efficiency of the new compression and restoration algorithms will also be tested and the algorithms will be evaluated.

1.4 Chapter Organization

Chapter 2 describes the background of the study by reviewing the main research issues in remote sensing and GIS, as well as in image processing and image compression. Fractal theories and applications are discussed; then the history of the fractal is introduced, including the rise of this new concept, the properties of the fractal, and other

advantages of its use, such as computer graphics, geography, and cartography. Finally, the combination of fractal concepts and image compression technique is considered.

Chapter 3 focuses on fractal image compression. First, the theoretical foundation is introduced: how fractals can be used in image compression and decompression; why fractal is expected to be better than other compression techniques; and how fractal compression is implemented with an effective image partition method, the quadtree method. Based on the foundation, the compression and decompression algorithms are developed and described.

Chapter 4 describes the research design. It starts with the description of study areas, which are located in the state of Louisiana and include a variety of landscapes, swamp, woodland, coastal line, gulf, and larger water bodies. The remote sensing data used in this research, Landsat TM images, are described.

Chapter 5 documents the processing results and analysis. The results are listed including the compression rates, computation times, and the comparison of the original and reconstructed images. It shows how different land covers/landscapes and pixel resolution can influence the compression rate, the computational time; the fidelity, the relationship between complexity of geographical phenomena and fractal dimensions, and the efficiency of the algorithms, are observed and analyzed. The result of fractal compression is also compared with the results of two most commonly used compression methods: WinZip and JPEG. The three research hypotheses are also tested.

Chapter 6 provides the conclusion and discusses the future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Development of Remote Sensing and GIS

Remote sensing and geographic information systems (GIS) are inherently linked technologies. Together, they form a powerful tool to measure, map, monitor, and model resources and environmental data for both scientific and commercial applications. Both remote sensing and GIS technology are used to collect, analyze, and report information about the earth's resources and the infrastructure (Star et al. 1997). The two technologies provide complementary capabilities. Remote sensing date analyses are improved by the verified data retrieved from a GIS, and GIS applications can benefit from the information that remote sensing can generate. Often the image data are the most current spatial information available for an area. The use of digital image data offers the additional advantage of a computer compatible format that can be input directly to a GIS (Foresman 1998).

Remote sensing and GIS technology developed independently. In part this was a result of the use of different types of equipment and the need for different technical skills. While a user of remote sensing technology may develop expertise in sensor systems and image processing methods, the expert GIS user may become more familiar with principles of map projections, spatial analysis, and the design of spatial data bases. Although the technology may encourage different technical orientations, in both cases the user must understand the nature of the information being collected: the vegetation, landscape, land use/land cover, geological features, urban areas, building structures, roadway design, and so on (Lillesand and Kiefer 1999).

Both remote sensing and GIS have common historical roots. In their initial development, digital GIS lagged behind digital remote sensing by more than a decade, but now the applications of GIS have far outstripped the applications of digital remotely sensed data in the market place. Perhaps the lag of GIS in development was the result of the lack of available digital data that could be input into GIS. However, now the development of remote sensing has provided great conditions for the development of GIS. GIS, on the other hand, requires more accurate, multi-scale, larger volume, and diversified geographic data. Remote sensing imagery can be one of the best sources; therefore both of them are driving each other's development (Ehlers 1990).

The integrated use of remote sensing and GIS methods and technology can not only improve the quality of geographic information but also enable information previously unavailable to be economically produced. Over the past two decades, manufacturers have developed more sophisticated technology for integrating remote sensing systems and geographic information systems (Star et al. 1997). For example, in the GIS software packages, such as Arc/Info, Intergraph's GeoMedia, and ERDAS, which is also an image processing system, remote sensing imagery can be directly used as the data source (Lidoy et al. 2000; InterGraph 2000; Esri 2001). It is more convenient for the users to take advantage of the combination of the two technologies.

The development of remote sensing and GIS has led to progress in several fields: the variety of applications; the theoretical research and analysis; and the techniques in remotely sensed image processing and analysis. Image compression, which is an important part of image processing technique, is the central part of this dissertation

research. In the following sections, the corresponding areas will be described based on the need of the study.

2.1.1 Applications

As the acquisition of information about an object without being in physical contact with it (Colwell 1983), remote sensing technique has been developing very rapidly in the past two decades. We are observing that the advent of the space age opens a whole new dimension in our ability to observe, study, and monitor planetary (including Earth) surfaces and atmospheres on a global and continuous scale. This has led to major developments in the field of remote sensing, both in its scientific and technical aspects. In addition, recent technological developments in detectors and digital electronics opened the entire electromagnetic spectrum. The full spectrum is useful in detecting and measuring even finer details of the emitted and reflected waves that contain the "fingerprints" of the medium with which they interact. Space-borne imaging spectrometers from the visible to the far infrared are being developed to acquire laboratory quality spectra for each observed surface pixel or atmospheric region, thus allowing direct identification of surface or atmospheric composition (Elach 1987).

Remote sensing is defined, in the application's point of view, as the measurement of object properties on the earth's surface using data acquired from aircraft and satellites. It is therefore an attempt to measure something at a distance, rather than in situ. While remote sensing data can consist of discrete, point measurements or a profile along a flight path, we are most interested here in measurements over a two-dimensional spatial grid, i.e. images. These images are the original first-hand data on which we apply a variety of processing techniques, then extract information for different applications. Remote

sensing systems, particularly those deployed on satellites, provide a repetitive and consistent view of the earth. This "earth view" is invaluable for monitoring the earth's system and the effect of human activities on the earth. Some of the important applications of remote sensing technologies are (Schowengerdt 1997):

a) Environmental assessment and monitoring:
    i) Including data analysis about human activity influencing the environment, such as the construction and operation of:
        (1) Highways
        (2) Railroads
        (3) Pipelines
        (4) Airports
        (5) Industrial sites
        (6) Power plants, and
        (7) Transmission lines
    ii) Subdivision and commercial developments
    iii) Hazardous waste disposal operations
    iv) Timber harvesting, and
    v) Strip mining operations, etc.
b) Global change detection and monitoring:
    i) Including:
        (1) Atmosphere ozone depletion
        (2) Deforestation, and
        (3) Global warming, and others.

c) Agriculture, including: crop condition, yield prediction, and soil erosion.

d) Non-renewable resource exploration, including: the exploration of minerals, oil, natural gas, etc.

e) Renewable natural resources, including: wetlands, soils, forests, oceans.

f) Meteorology including: atmosphere dynamics, weather prediction.

g) Mapping including: topography, land use, civil engineering.

h) Military surveillance and reconnaissance including: strategic policy, tactical assessment.

All the applications cannot be achieved without accurate remotely sensed data. This requires progress in the technology of image processing, storing, transferring, and analyzing, as well as the development of GIS. GIS potentially allows the manipulation and analysis of large volumes of data in ways that would be too costly, too time-consuming, or practically impossible to do using manual methods.

2.1.2 Research on Integration of Remote Sensing and GIS

GIS potentially has several definitions that define it in terms of its function as a component to various applications. Aronoff (1986) defines a GIS by stating that it "is designed for the collection, storage, and analysis of objects and phenomena where geographic location is an important characteristic or critical to the analysis". In the 1980's, GIS was still a combination of maps and spatial databases containing the attributes of the locations and describing the maps, although it had obtained great development as compared with earlier GISs. Along with the development of computer, communication, and remote sensing technology, as well as the requirements from applications, GIS is playing an increasingly important role in geo-science. Its functions

are also becoming broader. Not only can GIS store, update, and retrieve geographic data, and display simple map overlay and relational models, but also provide spatially distributed situation models of geographic phenomena and be used as a Spatial Decision Support System (SDSS). Dobson (1993)'s definition reflects this functionality: GIS is defined as "a digital representation of the landscape of a place (site, region, or planet), structured to support analysis.

The National Center for Geographic Information and Analysis (NCGIA) listed twelve research initiatives (Star et al. 1997):

1) Accuracy of Spatial Databases
2) Languages of Spatial Relations
3) Multiple Representations
4) Use and Value of Geographic Information in Decision Making
5) Architecture of Very Large Spatial Databases
6) Spatial Decision Support Systems
7) Visualization of the Quality of Spatial Information
8) Expert Systems for Cartographic Design
9) Institution Sharing Spatial Information
10) Temporal Relations in GIS
11) Space-Time Statistical Models in GIS
12) Integration of Remote Sensing and GIS Technologies

In (12), the relationship between remote sensing and GIS was further emphasized. Some areas, which involve data structure, data acquisition, data processing, data analysis,

and data conversion, were listed as the emphases. Research in these areas has obtained more attention since then.

Remote sensing employs a field-based model of space that is systematically sampled by the sensing system. GIS may be built from this perspective or from an object-oriented point of view. There are substantive differences in the levels of abstraction and precision between these representations. Consequently, transformations between the representations of geographic space need to be carried out. Transformations involve compromise in which spatial information is commonly lost; therefore, the information loss should be quantified and restricted within the "allowable" tolerance of this compromise (Star et al. 1997).

2.2 Image Compression

The goal of compression is to obtain an image representation while reducing as much as possible the amount of memory needed to encode the image (Lagonelijk and Biemond 1991). With the continuing growth of remote sensing technology, demand for image transmission and storage is increasing rapidly. Advances in computer technology for mass storage and digital processing have paved the way for implementing advanced data compression techniques to improve the efficiency of transmission and storage of images (Gomes and Luiz 1997). Image compression, as an important part of image processing technology, has been applied in some areas, such as visual signal transmission and communication. The research on this theory and methods has also been developed in past decades. Both the technology and research will be reviewed in the following sections.

2.2.1 General Concepts

Image compression maps an original image raster into a bit stream suitable for communication over or storage in a digital medium so that the number of bits required to represent the coded image is smaller than that required for the original image (Stored 1992). Ideally people would like the coded image to require as few bits as possible so as to minimize the storage space or communication time. They may also require in some applications that the original image be perfectly recoverable from the coded form. If the original image is an analog picture, it is improbable that the digitized information will be exactly the same as the analog original, no matter how many bits are used. However, digital to analog exists with great accuracy and therefore it is possible to convert analog to digital, but the instances of perfect conversion are not common. But people may face a series of issues. For example, the efficiency of compression algorithms, including data compressing rate, the resulting distortion, and its implementation complexity, is a particularly important considerations in hardware implementation and applications (Jain 1981, Bates and McDonnell 1985, Chen 1987, Abonsleman et al. 1995, Ratakonda and Ahuja 2002, and Saha 2002).

The basic goal of image compression is the conversion of an original sampled continuous or high bit-rate image into a "compressed" image with a binary coding having a specific amount of bits per pixel (bpp) so that the decompressed image has the best possible fidelity. The image decompression, or restoration, is the inverse procedure of compression: the compressed images are converted back to the original one, or the best approximation of the original images.

Before using a particular compression technique, it is important to know which category the image compression technique belongs to. There are two broad categories of image compression techniques. The first category consists of methods, which completely preserve the original data. When the compressed image is converted back into its uncompressed form, it is identical with the original image. This kind of technique is called “lossless” compression. For this kind of compression to be effective, there must be some redundancy in the original data. The second category of compression technique consists of methods that only approximate the original data. This category of compression is called “lossy” compression. In general, the less accuracy needed of the resulting image, the greater the compression rate (Roman 1997)..

Given these two categories of techniques, it is obviously important to know, for any given application, how much degradation in image quality can be tolerated. It is possible to apply objective measures of quality (Jahne 1993). For example, the original image can be compared with the result of compression followed by decompression. The differences can be expressed as a kind of signal to noise ratio, which can be used as a performance measure of the compression technique.

2.2.2 Performance Measures of Compression Methods

2.2.2.1 Entropy and Compression Rates

Entropy expresses the minimum number of bits necessary for the representation of an image without any loss of information. This is a global measure of the correlation between gray values of neighboring pixels (Novak and Shahin 1996).

For an n-bit image with M gray levels ($M=2^n$), the entropy H

$$H=-\sum_{i=1}^{n} p(g_i)\log_2 P(g_i) \quad (2\text{-}1)$$

where $P(g_i)$ is the percentage of occurrence of gray value.

$$H_{max}=\log_2 M \quad (2\text{-}2)$$

where M is the number of gray levels.

The redundancy R:

$$R = \log_2 M - H \quad (2\text{-}3)$$

Compression rate:

$$\Delta C_{max} = \log_2 M / H \quad (2\text{-}4)$$

2.2.2.2 Fidelity Measures

The techniques employed for image data compression may result in some degradation of the reconstructed image. So there should be a measure to judge the fidelity of the reconstructed image. Another widely used measure for an N x M size image is the mean-square error defined as

$$e^2_{ms} = (1/NM) \sum_{i=1}^{N} \sum_{j=1}^{M} E\,(u_{i,j} - u^*_{i,j})^2 \quad (2\text{-}5)$$

where $u_{i,j}$ and $u^*_{i,j}$ represent the N x M original and the reproduced images, respectively and E is the expectation value of the square of the difference between $u_{i,j}$ and $u^*_{i,j}$ .Experimentally, the mean-square error is often estimated by the sample mean-square error in the given image defined by:

$$e^2_{ms} \approx (1/NM) \sum_{i=1}^{N} \sum_{j=1}^{M} (u_{i,j} - u^*_{i,j})^2 \quad (2\text{-}6)$$

2.2.3 Image Compression Methods

The basic idea of image compression is to remove redundancy from the image. This is usually done through mapping the image to a set of coefficients. The resulting set

is then quantized to a number of possible values that are encoded by an appropriate coding method (Lammi 1995).

A basic image compression technique is to use an invertible linear transformation to transform the given correlated image into an array of uncorrelated variables that can be represented with fewer numbers of bits in storage compared to the original image. The best invertible transform is the one that produces uncorrelated transform coefficients. There are some linear transforms in the literature: Fourier transformation, cosine and sine transform, Hadamard transform, and Karhunen-Loeve transform (KLT) are all linear transforms. Among these, only the KLT can produce exactly uncorrelated transform coefficients for finite images, but the computational problems are serious. These include long computing time and large spaces, while other transforms, such as Fourier, cosine and sine, can be quickly carried out, but their coefficients are not 'exactly' uncorrelated. Moreover, the compression rates in the transforms are limited. So the results are not satisfactory (Eichmann et al. 1989, Gray et al. 1992, Hoffman and Johnson 1994, Lammi and Sarjakoske 1995, Chang 2002, Ratakonda and Ahuja 2002).

Predictive coding is another kind of compression method. Differential pulse code modulation (DPCM) is a predictive coding scheme (Shen 1989). The basic DPCM system consists of two main components: the predictor and the quantizer. The predictor uses the correlation between pixels to derive an estimate $g^*(i, j)$ for a given pixel value g (i, j) in terms of its neighboring pixels. The predicted value is removed from the actual value at the transmitter. The quantizer discretizes inputs at a specified interval, the prediction residual $h(i, j) = g(i, j) - g^*(i, j)$ to one of a finite number of values. The value is encoded and transmitted to the receiver. After decoding the transmitted codes, the

receiver reconstructs the pixel value by adding the predicted value to the quantized prediction error. Therefore, the success of a DPCM compression scheme depends on effective prediction of the current pixel value and efficient quantization of the prediction residual. In the algorithm, the image is divided into non-overlapping 8x8 neighborhoods starting from the upper left corner of the image. Each neighborhood is composed of 4 levels. The upper left corner pixel is a level 1 pixel, which is used to predict values of the remaining pixels through either linear or bilinear interpolation. The advantage of this method is "time-saving", but the disadvantage is that it is not stable. If the adjacent pixels have similar digital numbers, the result is good; if the values of pixels are much different, the result is not good.

The other type of image compression method is based on the discrete cosine transform (DCT) (Abonsleman et al. 1995). The JPEG algorithm proposed by the Joint Photographic Experts Group (JPEG) contains four modes of operations: sequential encoding; progressive encoding; lossless encoding; and hierarchical encoding, where the sequential and progressive encoding methods are based on the DCT while the lossless mode is based on a predictive method. The hierarchical mode encodes the image at multiple spatial resolutions using either the DCT-based compression or the lossless mode (Wallace 1991). The encoding methods of JPEG contain three sequential steps: forward discrete cosine transform (FDCT), quantization, and Huffman coding. The processing scheme is applied to a stream of 8 by 8 pixel blocks with 8 bits per pixel. Decompression is achieved by following the processing steps in the opposite direction: Huffman decoding, dequantization, and inverse discrete cosine transform (IDCT). The disadvantages of these methods include: decompression of large digital images is time-

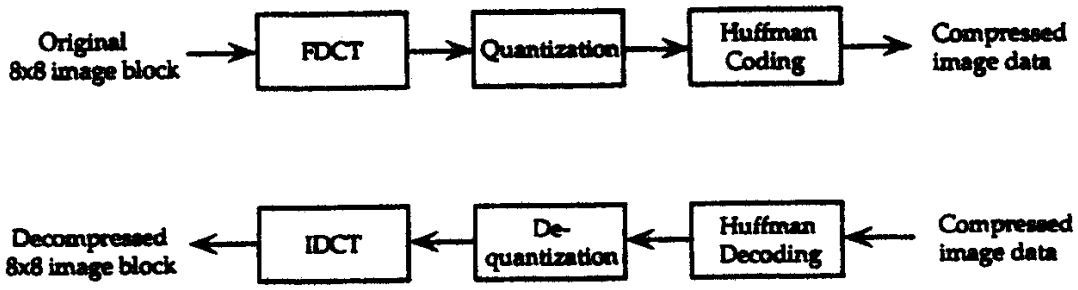


Figure 2.1 Baseline JPEG Compression Scheme (Source: Lammi and Sarjakoski 1996)

consuming; also some geometric degradation effect may occur with higher compression rates. Figure 2.1 shows the JPEG compression scheme.

2.3 The Fractal Model

"Fractal" is a new geometry proposed by French Mathematician Mandelbrot in the middle of the 1970s, which is really a revolution in topological space theory and provides the possibility of describing and simulating objects and phenomena precisely in the natural world by using a series of new concepts and mathematical models.

Geometry is a branch of mathematics that deals with the shape of objects. But people often describe geometry as "uninteresting" or "monotonous". One of the reasons is that the traditional or classical geometries based on the Euclidean system cannot accurately describe natural objects, such as clouds, mountains, coastlines, etc., because "clouds are not a sphere, mountains are not cones and coastlines are not circles" (Mandelbrot, 1983). The common feature of the shapes of most objects in the natural world is that they are neither regular, nor smooth. All the traditional geometries deal with regular and smooth geometrical shapes. In fact, the classical geometries, which are thought as "strict and accurate", are just inaccurate descriptions of the objects in the natural world; for example, the surface of the earth is always treated as an absolutely smooth spherical surface, or, ellipsoid in an "ideal situation". The value of Euclidean Geometry is dependent upon its use. In many instances it describes reality with great clarity. However, along with the progress of both science and technology, and mankind's knowledge about the world, people have found that Euclidean Geometry is not always useful. A geometry that accurately reflected niche realities was necessary. Therefore,

when Fractal Geometry was introduced, it drew many scientists' attention. This geometry found real world applications as soon as it appeared.

2.3.1 Development of the Theory

During a period of history, some people thought that a curve should be smooth along its entire path, or, at least smooth in recognized "segments", i.e. with few exceptions, every point has its tangent. In other words, in Euclidean Geometry, two points defined one line and only one line when transected through any of these two points Since only one line can pass through two points, these points are differentiable. There are only 2 linearly independent directions in a plane and only one on a line. Nevertheless, in the 19^{th} century, a German mathematician was the first person to find a continuous undifferentiable function. Later, Italian Mathematician Peano offered a curve which can fill an entire square but is undifferentiable everywhere. Subsequently, people found there are many such curves, now known as "Peano Curve" (Figure 2.2). As it is shown, in classical geometry, a point's dimension is zero; a line's dimension is one; while a plane's dimension is two, and a solid object's dimension is three, suggesting length, width, and height are three linearly independent directions. All dimensions in conventional geometry are integers. In other words, in classical geometry, all curves have a topological dimension D_T=1. Obviously the topological dimension D_T cannot reflect the features of fractal curves. So in fractal geometry, a new concept: *fractal dimension*, denoted as D, is used as the parameter to describe the feature of the curves:

$$D = log\ N / log\ (1/r) \qquad (2\text{-}7)$$

Where D is the fractal dimension; N is the number of segments with same length composing the curve and $1/r$ is the ratio of the distance between the two ends of the curve

and the length of the segment. The D of fractal curves is greater than 1, smaller than or equal to 2 (Figure 2.3). This is the feature of fractal curves, which is different from the "general curves". Also, based on the same principle, there are fractal surfaces with $2 < D \leq 3$ and fractal solid objects with $3 < D \leq 4$. So it can be shown that a fractal dimension is a measurement of the complexity of curves or surfaces: the more complex the curves or the surfaces, the greater the geometrical situation of objects based on its feature.

Central to the concept of fractals is the notion of self-similarity. Self-similarity means that for any curve or surface a portion of the curve or surface can be considered as a reduced image of the whole. In other words, the curves or surfaces are self similar to the object and makes up copies of itself in a reduced scale (Lam and Quattrochi, 1992). In the natural world, however, strict self-similarity seldom occurs. Therefore, the concept of statistical self-similarity is often applied.

2.3.2 Development in Applications

Since fractal theory was introduced, it has been applied to many fields of study, ranging from physics to music. The research of Brownian motion of small articles is an example. In computer graphics fractal theory is applied in many areas; generating complex curves or surfaces or simulating objects in the natural world is an illustration. In recent years, people find that not only is it useful in describing geometrical shapes, but also in describing many natural phenomena: "Fractal Everywhere" (Barnsley 1988). The applications of the fractal are briefly described below.

As mentioned above, the fractal's importance originates because of its accurate description of certain realities, including the measures of some geographical phenomena,

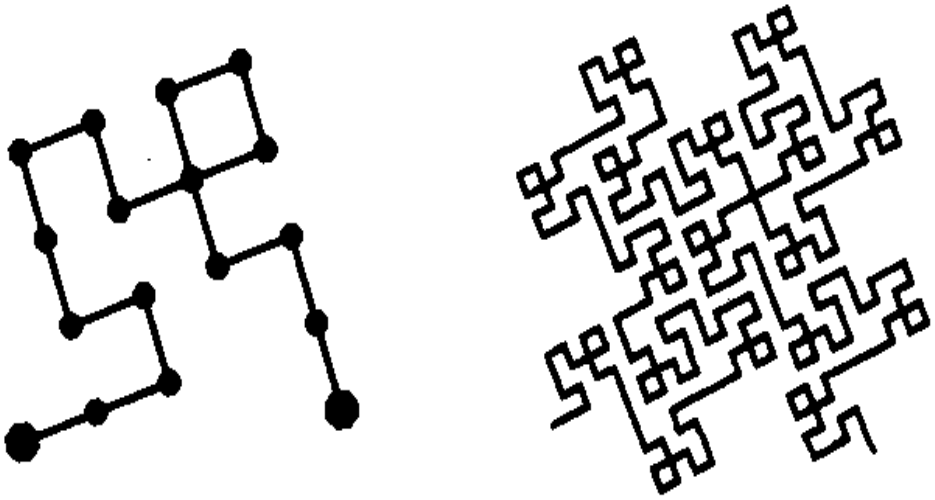


Figure 2.2 Peano Curve
(Source: Mandelbrot 1983)

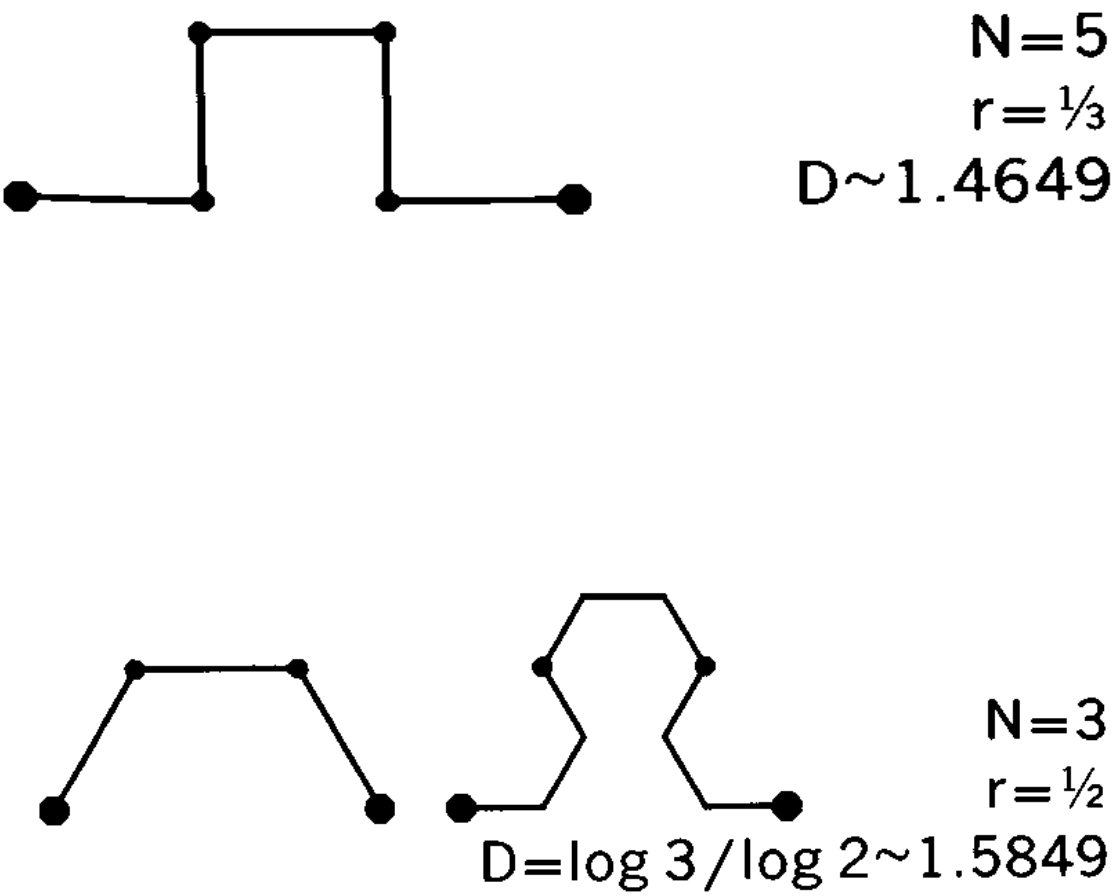


Figure 2.3 Fractal Dimensions
(Source: Mandelbrot 1983)

such as the length of a coastline. In 1967 Mandelbrot proposed the new concept used to capture and analyze the "new" information that traditional science could not provide. He asked, "How long is the British coastline?" and advanced the way to measure it by using the fractal concept. After the new geometry appeared, it has been attracting more and more attention by scientists. In 1980, Goodchild described the close relationship between fractal and geographical measures and pointed out that the fractal dimension can be used to predict the effect of cartographic generalization and spatial sampling (Goodchild 1980). Later on, Shelberg, Lam and Mollering advanced the algorithms using fractals to measure the fractal dimension of curves (Shelberg et. al. 1982) and surfaces (Shelberg et al. 1983), which is a meaningful beginning of applying the concepts of fractal dimension to measure the complexities and irregularities of the shape of curves and surfaces of "real objects". In cartography, Dutton used the properties of irregularity and self-similarity of the fractal to develop an algorithm to enhance the detail of digitized curves by altering their dimensionality in parametrically controlled self-similar fashion.

In other areas, other scientists made effort to use fractal in computer graphics. Batty (1985) showed a number of examples of simulated landscape, mountainscape and other graphics generated by using the property of "self-similarity" of fractals and pointed out the potential of development of applying fractal in this area.

2.3.3 Development in Geography

The research and application in geography mainly include physical geography, human geography, and mapping science. Below is a brief description of some applications documented in the book titled "Fractal in Geography" (Lam and DeCola 1993).

In his research in physical geography, Burrough (1993) introduced both fractal and geostatistical methods in landscape studies. The work included the methods of determining fractal dimensions of islands and lake outlines and other two-dimensional fractal shapes; the applications in geochemistry, such as estimating mineral reserves and pollutant levels, and applying fractals to porous media, such as caves and coral reefs; and using geostatistics to prove the fractality in the spatial variation of the properties of soil and rocks. Goodchild and Klinkenberg (1993) described the work of using fractal Brownian surface to simulate channel networks. Phillips (1993) worked on interpreting the fractal dimension of river networks and analyzing the relationships between fractal and classical fluvial morphometrical methods. Lavallee (1993) and others have their results in nonlinear variability of landscape topography. They used the concept of "multifractal", in geographic and geophysical fields, to analyze the properties of multifractal measures and simulate the landscape topography, making a comparison of mono and multifractal exponent relations. Plotnick and Prestegaard (1993) concentrated on fractal analysis of geologic time series; then they gave an example of analyzing sediment transport in gravel bed streams by using fractals, including the whole procedure from data collection to data analysis.

In research and applications in human geography, Arlinghaus (1993) worked in an urban setting with fractals used in "central place theory" in theoretical geography and pointed out the future development in this field. Lam (1993) and others applied the fractal to analyze cancer mortality patterns including esophageal cancer, stomach cancer, and liver cancer in the Jiangsu province, southeast of China. They compared different fractal measurement methods.

In research and applications in mapping sciences, Clarke (1993) used fractal to simulate the terrain with a digitized data map of Mountain Everest. He also used Fourier transformation and the inverse Fourier transformation to generate the simulation. Cola (1993) then gave a detailed description of using multi-fractals in image processing and process imaging, including multiscale analysis simulation and extraction of information from satellite images.

Based on the study and research of many scientists in geography and cartography, Goodchild and Mark (1993) reviewed the application of the fractal in three headings: the response of measure to scale; self-similarity; and recursive subdivision of space. They specifically pointed out that the self-similar property of fractal surfaces generated by using fractal Brownian motion makes them useful as null hypothesis landscapes in the study of geomorphology. Then they concluded that fractals should be regarded as a significant change in conventional paradigms about spatial form and as providing new and important standards of spatial phenomena.

Scientists working in remote sensing and the GIS area were also making an effort to put the fractal into practical use. Lam (1990) found that different land types have different levels of fractal dimension in different bands, and compared the fractal model with other spatial techniques and spatial statistics. Lam suggested that fractal dimension could be used to characterize remotely sensed data about land resources, land cover types, and landscapes. The practical use also included the measure of the fractal nature of the Louisiana coastline (Lam and Qiu 1992), such as the length of the base line from which the boundary of the state should be drawn. This involved detailed measures and interpretations. Based on these results, this conclusion followed: length, and other

cartographic measurements are related to scale; and scale and resolution, as well as fractals play an important role in geography. Lam and Quattrochi (1992) pointed out that the optimal scale and resolution for a study depend on the need for the study, the type of environment, and the type of information desired. Fractal dimension can be used to summarize the scale changes of spatial phenomena. In another research project Lam and her group compared the fractal model with other spatial techniques and spatial statistics in characterizing remotely sensed data for land resource and landscape interpretation (Quattrochi et al. 1997).

In Lam and De Cola (1993), the authors described the fundamental concepts of the fractal, and the fractal measurement of geographical objects, including the methods for measuring curves and surfaces. They also showed other uses of the fractal: simulation, such as Carpenter's midpoint subdivision and Dutton's midpoint displacement method for curves and Shear displacement method for surfaces as well as some other methods.

In addition to the basic concepts and some applications, the authors summarize "a fractal paradigm for geography." They explained the relationship among geography, geometry, and cartography. They led the readers on a tour in the areas of fractal nature and "close" to geography, such as astronomy, and geology. This helped the readers understand fractal. They discussed future work in geographical areas, including improving the fractal techniques, combining it with other techniques and developing the theory itself.

There was also some work done in the remote sensing area, such as multiscale remote sensing analysis, including the research of how scale and resolution change image

information and also how fractal can be used to reflect this scale (Lam 1992). In addition, research determines how the fractal model is compared with other spatial techniques, as well as spatial statistics in characterizing remote sensing data. This includes land resources, land cover types, and landscapes, as well as describing and measuring Landsat TM image (Lam 1990); analyzing the fractal nature of Louisiana coastline (Lam et al. 1992) and other meaningful work involving the fractal concept.

2.4 Considerations

Fractal, the new science and technique, has been developed greatly since it was first advanced. But it is still new and also immature in both theoretical and practical areas. So it can be expected that more development would be obtained in the future.

Along the direction of the existing work, there are some areas to develop:

- Fractal theory is still descriptive rather than inferential. It provides some models to simulate landscapes, but the models do not provide people with something more than descriptive tools. So the theory should be developed to meet the demands of applications. New methods or techniques also need to be tested and improved. On the other hand, the fractal analysis of landscapes, as a part of the developing field of fractals, is also in its preliminary and descriptive stage, and lacks theoretical explanation. The relationship between the process underlying the landscape and its fractal characteristics remains to be explained. So besides the fractal theory, the research of geographical information theory, which is used to describe objects phenomena or processes at the earth's surface, also needs to be strengthened.

- It has been found that fractal dimension or self-similarity only exists in certain ranges of scale. So this property can be used to determine the scale at which the geographical phenomena or geomorphology can be better analyzed and the break points in a pattern or distribution can be identified where the processes contributing to these patterns become unstable. For example, a homogeneous pattern becomes heterogeneous and vice versa (Lam and De Cola 1993).
- Fractal dimensions provide the measure of the complexity of curves or surfaces, but it is only a "descriptor." In other words, with the fractal, one can only get the impression of how complex or how coarse the curve or surface may be. While Fourier Transformation can reconstruct or regenerate the curve or the surface with the Fourier coefficients, one cannot get the impression of complexity. These methods have their own advantages and disadvantages. They also have something in common. For instance, the more complex a curve or a surface, the higher the fractal dimension; while in Fourier Transformation, the higher the frequency in frequency domain. So the two techniques could be combined to make the results more precise and more reasonable.
- Image encoding and compressing is a very significant branch of image processing. The research is to organize the images and make them more compact by using the characteristics of images as information source and information carrier in order to extract useful information and store the images effectively, and when needed, restore them without losing useful information. So far there have been encoding and compressing methods available, such as raster encoding, run-length encoding and quadtree encoding and compressing. But, the problem is that the methods are

based on *the format* of the image, rather than *the individual features* of the images, so it is difficult to reflect the characteristics of the image accurately. This may interfere with the extraction of useful information and the effect of encoding and compressing. The other problem is that the compression rates of current methods for compressing are usually low. Generally speaking, the best result is to reduce the image size to 25 percent of the original image size. So in most cases, it may not solve the existing problem of storage.

As mentioned above, the fractal has received enormous development since it arose in the 1970s. Then again, the fractal has only about twenty years of history. It is a "developing" theory and technique; in other words, it needs to be developed in both theory and application.

It seems that Mandelbrot himself was involved in a paradox: he was trying to make a new model to replace the old one which can only deal with an "ideal situation," but what he advanced still deals with some other "ideal situation" which rarely, or even does not exist in the real world — pure fractal and absolute self-similarity. This may be the limitation, which is from the mathematician's points of view: to seek something perfect, ideal, and strict to meet the requirement of mathematics. But the real world is not exactly as the mathematicians imagine, it is chaotic rather than ordered. Mandelbrot made some modifications to fit his theory into the physical world. Mathematics as an internal construct creates its own reality. The practical engineer attempts to fit the mathematical model into his niche reality. For example, every structure ever built in the world could be built suitably using Euclidean geometry. Travel to the stars might necessitate a different approach to niche reality. Mandelbrot applied Fractal mathematics to his niche reality.

Undoubtedly the rise of fractal geometry is a breaking point in this scientific period, but it is not perfect. Issues demand attention and techniques that more clearly define niche reality in terms of fidelity and utility need to be developed. At this point, the cooperation between mathematicians and scientists in practical areas is urgently needed. The former can provide the latter with new theories or new developments in existing theories. The latter can provide the former with experimental data and the demands of practical use. With cooperation, more development of both fractal theory and application can be expected. Using the concepts of fractals in the practical application of image compression is a needed area of research. In the next chapter, fractal image compression is introduced and described in detail.

CHAPTER 3

FRACTAL IMAGE COMPRESSION

This chapter describes the fractal compression scheme. Image compression is a very significant and necessary branch of image processing. The subsequent goal of image compression is to organize the images so they are more compact by using the characteristics of images as an information source and an information carrier. By organizing the images as effectively and compactly as achievable, useful information can be accurately extracted and the images stored effectively. When needed, these images can without difficulty be restored and they will remain accurate and will not lose functional or constructive information.

To this point, there have been some less effective encoding and compressing methods available. But, the former methods generate in their own fashion a difficulty that interferes with efficiency of use and storage. Primitive former methods are the product of their design, which did not include geography and the inherent needs of this discipline; therefore, these methods were primarily based upon the format of the image. This format does not factor into the equation the individual features of the images and their special peculiarities. Per se, it is inherently problematical to imitate the characteristics of the original image accurately for the purposes discussed in this context. As a result, dated data compression schemes potentially and in actuality interfered with the extraction and utilization of the valuable information. Thus the efficiency of realistically encoding and compressing images was de facto abridged. The second associated difficulty arising from dated methods of compression and analysis is that the compression rates of existing methods for compressing images are usually lower than tolerable. In the case of remotely

sensed imagery, the problem becomes even more profound. Therefore it is imperative to develop new and efficient compressing algorithms to solve the problem and meet the need of applications. Fractal methods provide the potential of accurately and effectively solving the stated problems.

3.1 Basic Concepts

3.1.1 Attractor and Self-similarity

Fractals have provided a different description of objects, which is completely different from traditional concepts. Fractal theory concentrates on the "self-similarity" of objects, so it can theoretically be used to describe the *inside* features of different images rather than something on the "*surface*". Therefore the route of compression by means of the fractal is different from traditional compression methods. An intuitive example would be a photocopying machine (Fisher 1992). Imagine a copying machine that reduces the image to be copied by one half and reproduces it three times on the copy tray (Figure 3.1). When we feed the output of this machine back as input, we will find that all the copies seem to be converging to the same final image after several iterations of this process on several input images (Figure 3.2). This final image is called the *attractor* for this copying machine. Because the copying machine reduces the input image, *any* initial image will be reduced to a point as we repeatedly run the machine. Thus no matter what initial image is placed on the copying machine, it will not affect the final attractor. In fact, only the position and the orientation of the copies determine what the final image will look like.

From the copies, it can be seen that each copy is formed of three reduced copies of itself. A common feature of these reduced copies and all attractors formed this way is that in the position of each of the images of the original square there is a transformed copy of

the whole image. So each image is formed from transformed (and reduced copies) of itself and hence it has the same detail at every scale. This is "*self-similarity*", the unique and important feature of fractal.

3.1.2 Contractive Transformations

Since the way the input image is transformed determines the final result of running the copy machine in a feedback loop, the transformation will be described. Different transformations will lead to different attractors (final images), provided that the transformations must be *contractive*, that is, given a transformation W, any two points P_1, P_2, in the input image must be closer in the copy. In other words, the distance between the two points:

$$d(W(P_1), W(P_2)) < s\, d(P_1, P_2) \tag{3-1}$$

for s < 1. In the case of a plane, if the points have coordinates $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, then

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{3-2}$$

This condition is natural and obvious, because if the transformation is not contractive, points in the copy will be spread out. It follows that the final image will be of infinite size. Except for this condition, the transformations can have any form. In practice, the transformation can be *affine*, which will be sufficient to yield an interesting set of attractors:

$$w_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \tag{3-3}$$

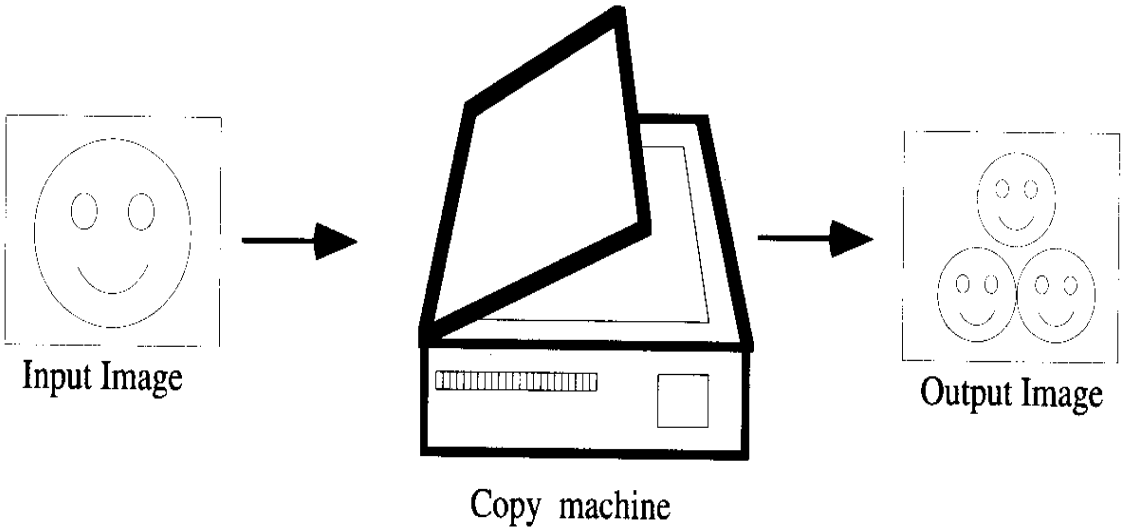


Figure 3.1 Copy Machine
(Modified from Fisher 1992)

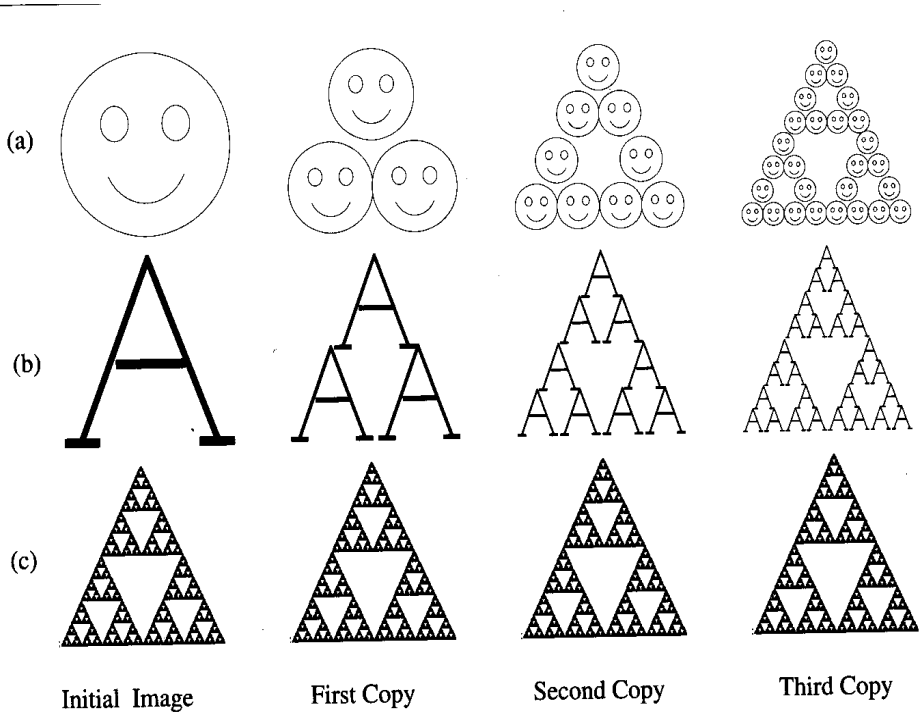


Figure 3.2 Different Initial Images with Similar Final Images
(Modified from Fisher 1992)

where w_i is the transformation; x and y are the coordinate of a point in the image; a_i, b_i, c_i, d_i, e_i, and f_i are the coefficients of the transformation.

Contractive transformations have the pleasing property such that when they are repeatedly applied, they converge to a point. This point remains fixed when further iterations are applied.

3.1.3 Iterated Function Systems (IFS)

An *iterated function system* (IFS) consists of a collection of *contractive transformations* $\{w_i: R^2 \rightarrow R^2 \mid i = 1, 2, \ldots, n\}$ which map the plane R^2 into itself. This collection of transformations defines a map:

$$W(\cdot) = \bigcup_{i=1}^{n} w_i(\cdot) \qquad (3\text{-}4)$$

where $W(\cdot)$ denotes the map, which consists of a set of transformations $w_i(\cdot)$; $(\cdot)$ is a group of points in the plane R^2. The map W is not applied to the plane, but as an alternative, it is applied to sets, collections of points in the plane. The special copy machine mentioned above, which is running in a feedback loop, is a good metaphor for IFS: Given an input set S, we can compute $w_i(S)$ for each i (corresponding to making a reduced copy of the input image S), and take the union of these sets (corresponding to assembling the reduced copies). Then W is a map on the space of subsets of the plane, in other words, a map on the space of images, not on the whole plane R^2 (Fisher 1992).

If a contractive map W on a space of image exists, then there is a special image, called the *attractor*, denoted as x_w, with the properties:

1. If the copy machine is applied to the attractor, the output is equal to the input. The image is fixed, and the attractor x_w is called the *fixed point* of W. Then we have:

$$W(x_w) = x_w = w_1(x_w) \cup w_2(x_w) \cup \ldots \cup w_n(x_w) \quad (3\text{-}5)$$

2. Given an input image S_0, we can run the copy machine once to get $S_1 = W(S_0)$; twice to get $S_2 = W(S_1) = W(W(S_0)) \equiv W^{o2}$, and so on. The superscript "o" indicates that we are using iterations, not exponents; for that reason, W^{o2} is the output of the second iteration. The attractor, which is the result of running the copy machine in a feedback loop, is the limit set

$$x_w \equiv S_\infty = \lim_{n\to\infty} W^{on}(S_0) \quad (3\text{-}6)$$

which is not dependent on the choice of S_0.

3. x_w is unique. If we find any set S and an image transformation (map) W satisfying $W(S) = S$, then S is the attractor of W; that is, $S = x_w$. It means that only one set will satisfy the fixed-point equation (3-5).

3.1.4 Partitioned Iteration Function System

Theoretically, each image has a unique fixed point. But in practice, it is impossible to find a unique fixed point for a whole image. Thus, the image should be partitioned as different parts, and the fixed points for the corresponding parts should be obtained through different transformations (or maps). The above IFS will be different from different parts of the image, and therefore it is called a partitioned iteration function system (PIFS). The copy machine metaphor can be extended to describe the process. Besides the copy machine's features described above, which include:

- The number of copies of the original pasted together to form the output;
- A setting of position and scaling, stretching, skewing, and rotation factors for each copy.

The extended one has more:

- A contrast and brightness adjustment for each copy;
- A mask that selects, for each copy, a part of the original to be copied.

The new features are applied to allow the transformation of gray-scale images. In particular, the final listed feature allows partitions of an image into pieces which are each transformed separately.

What happens when we copy an original image using this machine? A portion of the original, which is denoted as D_i, is copied to a part of the produced copy, denoted as R_i. The D_i and R_i are called *Domains* and *Ranges* respectively. The transformation between D_i and R_i is denoted as W_i. As before, the copy machine runs in a feedback loop: its own output is fed back as its new input repeatedly. The extended copy machine can be called a partitioned copy machine. The mathematical analogue of a partitioned copy machine is a *partitioned iterated function system*. Using affine transformation on the gray scale images, the gray level adds another dimension, so the transformation w_i will take the format like (Fisher 1992):

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \tag{3-7}$$

where s_i controls the contrast and o_i controls the brightness of the transformation. We can have the equation similar to (3-3):

$$v_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \tag{3-8}$$

An image is modeled as a function $f(x, y)$, so the transformation w_i is applied to an image f as $w_i(f) \equiv w_i(x, y, f(x, y))$. Then v_i determines how the partitioned domains of an original are mapped to the copy, while s_i and o_i determine the contrast and brightness of the transformation (Fisher 1997).

3.2 Theorems Related to Fractal Image Compression

As early as in 1988, Michael Barnsley began to research the issue of image compression by using the fractal concept (Barnsley and Sloan 1988). Then he and another scientist proposed a series of theorems to serve as the theoretical basis of fractal image compression (Barnsley and Hurd 1993). The basic idea of the theorems is that in order to find an IFS (Iterated Function System) whose attractor is close to, or looks like a given set, people should seek to find a set of contractive transformations on a suitable space with which the given set lies, such that the union, or collage of the images of the given set under the transformation is close to or looks like the given set (Fisher et al. 1992). The kernel is the contractive mapping fixed point theorem and collage theorem. Based on their theories, the algorithm of image compressing is developed and applied in this study.

3.2.1 Contractive Mapping Fixed Point Theorem

The contractive mapping fixed point of transformations theorem says that something that is intuitively obvious: if a map is contractive then when we apply it repeatedly starting with any initial point we converge to a unique fixed point. For example, the map $w(x) = ½\, x$ on the real line is contractive for the normal metric $d(x, y) = | x - y |$, because the distance between $w(x)$ and $w(y)$ is half the distance between x and y. Furthermore, if we iterated w from any initial point x, we get a sequence of points $½\, x$, $¼\, x$, … that converges to the fixed point 0.

This simple sounding theorem tells us when we can expect a collection of transformations to define an image. The precise format of the theorem is:

If X is a complete metric space and W: X → X is contractive, then W has a unique fixed point |W|.

What do these terms mean? A *complete metric space* is a "gap-less" space on which we can measure the distance between any two points. For example, the real line is a complete metric space with distance between any two points x and y given by $| x - y |$. The set of all fractions of integers, however, is not complete. We can measure the distance between two fractions in the same way, but between any two elements of the space we find a real number (that is, a "gap"), which is not a fraction and hence is not in the space. Returning to our example, the map w can operate on the space of fractions; however, the map $x \rightarrow (1/\pi)\, x$ cannot. This map is contractive, but after one application of the map we are no longer in the same space at which we start. This is one problem that can occur when we do not work in a complete metric space. Another problem is that we can find a sequence of points that do not converge to a point in the space; for example, there are sequences of numbers that get closer and closer to $\sqrt{2}$, which are not ordinary or decimal fractions.

A *fixed point* $|W| \in X$ of W is a point that satisfies $W(|W|) = |W|$. The mapping $w(x) = \frac{1}{2}\, x$ on the real line has a unique fixed point 0 because $w(0) = 0$ (Fisher 1992).

Proving the theorem is as easy as finding the fixed point: start with an arbitrary point $x \in X$. Now iterate W to get a sequence of points x, $W(x)$, $W(W(x))$, …. How far can we get at each step? The distance between $W(x)$ and $W(W(x))$ is less by some factor $s < 1$ than the distance between x and $W(x)$. So at each step the distance to the next point is less by some

factor than the distance to the previous point. Since we are taking geometrically smaller steps, and since our space has no gaps, we must eventually converge to a point in the space which we denote $|W| = \lim_{n\to\infty} W^{on}(x)$. This point is fixed, because applying W one more time is the same as starting at $W(x)$ instead of x, and either way we get to the same point.

The fixed point is unique because if we assume that there are two, then we will get a contradiction: Suppose there are two fixed points x_1 and x_2; then the distance between $W(x_1)$ and $W(x_2)$, which is the distance between x_1 and x_2 since they are both fixed points, would have to be smaller than the distance between x_1 and x_2; this is a contradiction.

Thus, the main result is that when W is contractive, we get a fixed point

$$|W| = \lim_{n\to\infty} W^{on}(x)$$

for *any* initial x.

3.2.2 Collage Theorem

This theorem is central to the fractal image compression methodology. It states that in order to find an IFS whose attractor is "close to" or "looks like" a given set, one must endeavor to find a set of transformations - contraction mappings on a suitable set within which the given set lies - such that the union, or **collage**, of the images of the given set under transformations is near to the given set. It is illustrated in diverse methods. Here we use Fisher's (Fisher 1997):

Collage Theorem: With the hypothesis of the Contractive Mapping Fixed point Theorem,

$$d(x, x_f) \le \frac{1}{1-s}\, d(x, f(x)) \qquad (3\text{-}9)$$

where x is any point in X, and X is a complete metric space; $f : X \rightarrow X$ is a contractive mapping (transformation), and x_f is the fixed point belonging to X.

Fisher's proof of the *contractive mapping fixed-point theorem* is beautiful and straightforward. It can be easily used as the proof of this theorem:

Select x belonging to X, then for n > m, existing $s < 1$

$$d(f^{om}(x), f^{on}(x)) < s\, d(f^{om-1}(x), f^{on-1}(x)) < s^m\, d(x, f^{on-m}(x)) \qquad (3\text{-}10)$$

using (3-10) and the triangle inequality repeatedly,

$$\begin{aligned} & d(x, f^{ok}(x)) \le d(x, f^{ok-1}(x)) + d(f^{ok-1}(x), f^{ok}(x)) \\ \le\ & d(x, f(x)) + d(f(x), f(f(x))) + \cdots + d(f^{ok-1}(x), f^{ok}(x)) \\ \le\ & (1 + s + s^2 + \cdots + s^{k-2} + s^{k-1})\, d(x, f(x)) \\ \le\ & \frac{1}{1-s}\, d(x, f(x)) \end{aligned} \qquad (3\text{-}11)$$

Taking the limit as k goes to infinity in (3-11), and rewrite (3-6), we have:

$$x_f = f(x_f) = \lim_{k \rightarrow \infty} f^{ok}(x)$$

then the theorem is proved.

3.3 Procedure of Fractal Image Compression

3.3.1 General Procedure

The image compression procedure is in fact an image encoding process while the reconstructing or restoring procedure corresponds to image decoding. For simplicity, monochrome images are considered here. The multi-spectral images are actually the

combination of separated monochrome images. As a result, the same way to deal with monochrome images can be extended into the situation of multi-spectral images (Jacquin 1993). Let (M, d) denote a metric space of digital images, where d is the distance measure (which is invariant to translation and rotation) in the given space. Let U be an original image that we want to encode or compress. We need to find a contractive image transformation T, defined from the space (M, d) to itself, for which U is an approximate fixed point. So, there exists a number $s < 1$ such that for any u, v belonging to (M, d), we have

$$d(T(u),T(v)) \leq sd(u,v) \tag{3-12}$$

and

$$d(U, T(U)) \approx 0 \tag{3-13}$$

The scalar s is called the contractivity of the transformation T. By repeated application of the triangular inequality in (M, d) and user of the contractivity of T, it is easy to have:

$$d(U, T^n(U_0)) \leq \frac{1}{1-s} d(U, T(U)) + S^n d(U, U_0) \tag{3-14}$$

then, from (3-1) and (3-2), and since $s < 1$, after a number of initial iterations, we will have:

$$\{U_n = T^n(U_0)\}|_{n \geq 0} \tag{3-15}$$

where U_0 is an *arbitrary* initial image. The sequence of U_n converges to a stable image, as the result of its iterative construction. So if a suitable T is chosen, it can be expected that the original image will be reconstructed as close approximation through a series of T

transformations from an initial image, which could be much simpler than the original image itself. In other words, after a series of *T* transformations, the initial image will converge to the original one. Reconstruction can be accomplished by partitioning the original image in a suitable manner that will obtain the suitable transformation *T* to meet the requirement of compression rate and computational issues. So the task of compressing an image includes three important parts:

1) Partition the image and find transformations for each partitioned part;
2) Encoding (compressing) the image; and
3) Decoding (decompressing) the image.

3.3.2 Partitioning Images

It is crucial to choose a suitable way of partitioning images or a suitable image data structure, because the method of partitioning images is closely related to the next step: algorithm development issue. The format of the data determines the manner of obtaining the data and the processing methods most effectively used upon the data. If an accurate method of partitioning image data is chosen for a specific issue, then the solution of the issue will be simplified. For the issue of fractal image compression, we choose the quadtree method to partition images because of its orderliness, simplicity, and efficiency.

3.3.2.1 Quadtree Partition

Quadtree is an image structure, which appeared at the end of the 1970s, and was developed and applied in the 1980s and the 1990s. In the first case, Klinger and Dyer (1976) advanced the concept of "evenly decomposing" in order to build a representation of complex image data designed for computer searching, and defined the corresponding

method of decomposition. The basic idea is to decompose images by region, rather than by rows or columns, with the intention that the structural information of images can be better reflected. One of the advantages of decomposing the image is: if an image is too big to be loaded into the available computer memory en bloc, this loaded image can be divided into sub-images to process without breaking the structural information of the image.

If we decompose an image into quadrants continuously, the result will be a quadtree. To imitate the definition of the binary-tree by Knuth (1980), we include the recursive definition about quadtrees:

> *A quadtree is a finite set consisting of several nodes which are either empty or consist of a root and at most four non-overlapping quadtrees.*

To get a quadtree of an image, the procedure of evenly decomposing the image is:

Suppose we have a 2^n x 2^n binary image in which points with value "1" represent the "feature" points as black ones and points with value "0" represent background points as white ones. First, the whole image is served as the root node. If the node does not consist of all "1" value or all "0" value points, it is called a gray node and needs further decomposition. As the first step, divide the whole image into four 2^{n-1} x 2^{n-1} sub-images, then decide if further decomposition is needed. The sub-images are not only son nodes of the whole image but also the root nodes of its own. If one node consists of all "1" value points or all "0" value points, then stop decomposing the node; if the node has both "1" value points and "0" value points, then decompose it again until all sub-images consist of points with the same values. (Figure 3.3)

In the very beginning, the quadtree took the format of the chain-structure and hierarchical style. Each node has six fields in which only one is with the value of the node and the other five are all chain fields pointing to the father node and the four son nodes of the current node. Also, since there are a lot of "gray" nodes as intermediate nodes in a quadtree, only ¾ nodes in total are black or white. In fact, only those points are "meaningful", more specifically, only black nodes are requisite for processing. Obviously, this kind of hierarchical quadtree occupies huge amounts of storage. Albeit, the structure is clear and simple: one fourth of the storage space is used to accumulate the intermediate nodes; and in each node, there are five out of six fields being used to store the pointers, and only one field to store the value of the node. The redundancy is very large; subsequently, the efficiency is very low. At this stage, the quadtree could not be put into practice mainly because of these disadvantages.

Gargantini (1982) advanced a new structure of quadtrees, called the "linear quadtree" (Figure 3.4). The differences of Gargantini's linear quadtree and the previous quadtrees are:

1) It only stores black nodes;
2) It codes each node;
3) The codes of nodes imply the path from root to nodes; and
4) Each region on an image can be represented as an ordered series of nodes.

The main advantages of linear quadtrees include:

1) The storage and processing time depend only on the amount of black nodes;
2) They remove the chain structure and a large amount of redundancy.

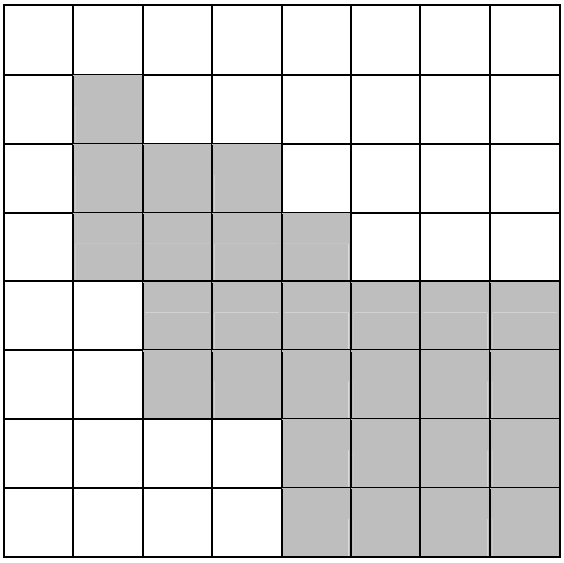
Based on Gargantini's work, Lauzon et al. (1985) introduced one coding method for linear quadtrees, called "Two Dimensional Run-Encoding", simply "2DRE". It takes advantage of the "Morton series" so that the codes of the quadtree structures become more compact (Figure 3.5).

At this stage, the quadtree structure became practical, applicable, and operational. Then the quadtree structure began to be used in image processing and image encoding areas (Xiao 1991, 1992, 1993). This method will be used as a way of partitioning images in this research.

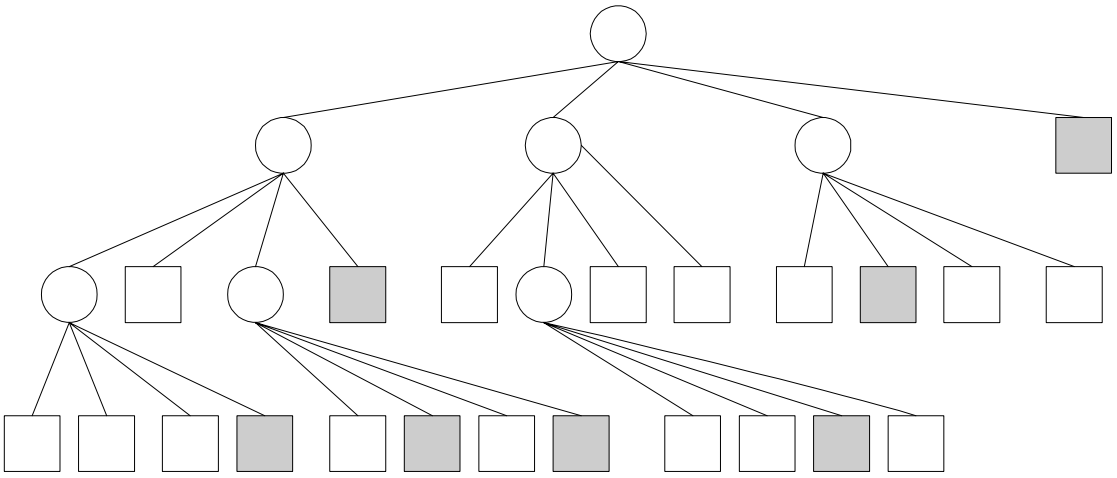
Based on the basic concepts and theoretical foundation that were discussed above, the process of image compression starts with image partitioning. As the first step, the image is partitioned by some collection of ranges R_i. Then for each R_i, a domain D_i, which has a low rms error, is found from some collection of image pieces. The ways of partitioning the image into domains and ranges are both quadtree method. The sets R_i and D_i, determine s_i and o_i as well as a_i, b_i, c_i, d_i, e_i, and f_i in (3-7). Then a transformation $W = \mathrm{U}w_i$, which encodes the original image, is obtained.

A quadtree partition is a representation of an image as a binary space partitioning tree in which each node, corresponding to a square portion of the image, contains four sub-nodes, corresponding to the four quadrants of the square. The root of the tree is the initial image (Figure 3.6).

First, the image is divided into domains with different sizes using the quadtree method and the domain pool is built, including sizes and positions of the domains. At that time, ranges are selected. Based on the minimum tree depth, various initial numbers of the quadtree partitions are prepared. The squares at the nodes are compared with



(a)



(b)

Figure 3.3 A Black-White Image and Its Quadtree
a. The Image
b. The Quadtree

domains in the domain pool (or domain library) D, which are twice the range size. The pixels in the domain are averaged in groups of four so that the domain is reduced to the size of the range, and the affine transformation of the pixel values is found that minimizes the rms difference between the transformed domain pixel values and the range pixel values. All the potential domains are compared with a range. If the resulting optimal rms value is above a pre-selected threshold and if the depth of the quadtree is less than a pre-selected maximum depth, then the range square is subdivided into four quadrants, and the process is repeated. If the rms value is below the threshold, the optimal domain and the affine transformation on the pixel values are stored. Thus one map w_i is found. The collection of all such maps $W = \mathrm{U}w_i$ constitutes the encoding.

3.3.2.2 Determining Parameters

Once the image is partitioned, we need to determine the transformation coefficients for each partition.

Given two squares containing n pixels intensities, $a_1, \ldots, a_n$ (from D_i) and $b_1, \ldots, b_n$ (from R_i). We can seek s and o to minimize the quantity

$$R = \sum_{i=1}^{n} (s \cdot a_i + o - b_i)^2 \qquad (3\text{-}16)$$

This will give us a contrast and brightness setting that makes the affinely transformed a_i values have the least squared distance from the b_i values. The minimum of R occurs when the partial derivatives with respect to s and o are zero, which occurs when

$$s = [\, n^2(\sum_{i=1}^{n} a_i\, b_i) - (\sum_{i=1}^{n} a_i)\,(\sum_{i=1}^{n} b_i)] \,/\, [n^2 \sum_{i=1}^{n} a_i^2 - (\sum_{i=1}^{n} a_i)^2] \qquad (3\text{-}17)$$

and

	003						
	021	030	031				
	023	032	033	122			
		210	211	300	301	310	311
		212	213	302	303	312	313
				320	321	330	331

Figure 3.4 Coding of a Linear Quadtree

	0	1	2	3	4	5	6	7	8
0	**0**	**1**	**4**	**5**	**16**	**17**	**20**	**21**	**64**
1	**2**	**3**	**6**	**7**	**18**	**19**	**22**	**23**	**66**
2	**8**	**9**	**12**	**13**	**24**	**25**	**28**	**29**	**72**
3	**10**	**11**	**14**	**15**	**26**	**27**	**30**	**31**	**74**
4	**32**	**33**	**36**	**37**	**48**	**49**	**52**	**53**	**96**
5	**34**	**35**	**38**	**39**	**50**	**51**	**54**	**55**	**98**
6	**40**	**41**	**44**	**45**	**56**	**57**	**60**	**61**	**104**
7	**42**	**43**	**46**	**47**	**58**	**59**	**62**	**63**	**106**
8	**128**	**129**	**132**	**133**	**144**	**145**	**148**	**149**	**192**

Figure 3.5 2DRE The quadtree with Morton Series Code
(The first 64 pixels and adjacent pixels of an image)

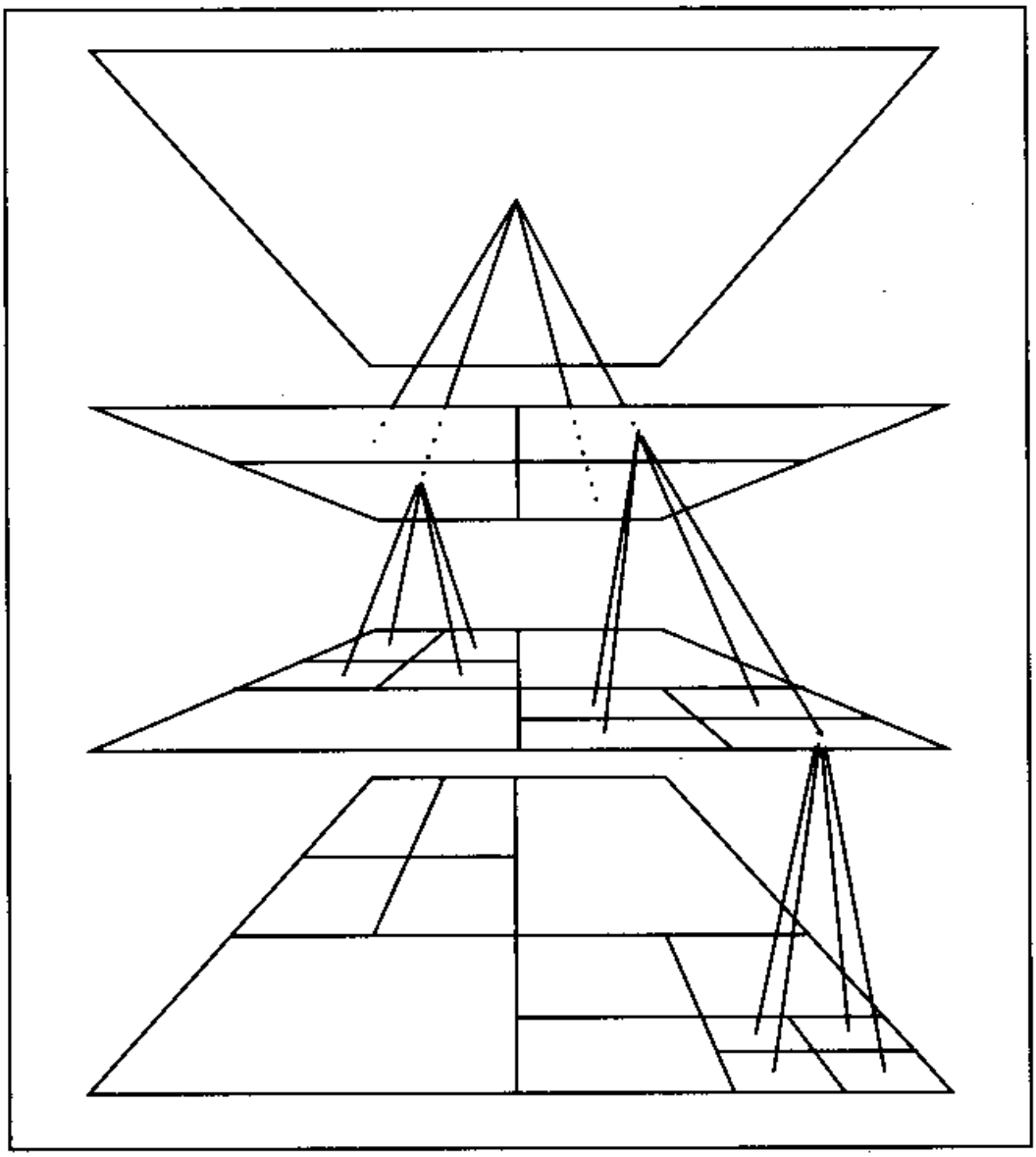


Figure 3.6 Quadtree Partition
(Modified from Fisher, Y., et al, 1992)

$$o = [\sum_{i=1}^{n} b_i - s \sum_{i=1}^{n} a_i] / n^2 \quad (3\text{-}18)$$

3.3.2 Encoding (Compressing) Algorithm

Based on the three key concepts described in the three theorems, the fractal image compression and decompression algorithms are developed in C code and implemented in this study. The image compression is named fractal because the method used to encode images shares many features in common with fractal generating algorithms (Fisher 1992). The basis for the encoding procedure is like this: an image is partitioned into parts that can be approximated by other parts after some scaling operations (Figure 3.7). The result of the procedure is a set of transformations, which, when iterated from *any* initial image, possess a fixed point approximating the original image. In the restored image, fractal characteristics can be seen: 'zooming' into the restored image, finer and finer details will appear. Accordingly, 'self-similarity' (a part of the image can be approximated by other part of the same image), and recursive subdivision, which are the important characteristics of fractals, are put into use in the encoding procedure. This is why the procedure is named 'fractal'.

The steps of the encoding (compressing) procedure are as follows:

1) Determine the parameters for compressing:
    i) Image name, image size, minimum partition exponent, maximum partition exponent (both of which determine the size of domains and ranges), tolerance for fidelity, e.g. xxx.img, 256x256, 4 (corresponding to 16x16 blocks), 6 (corresponding to 4x4 blocks), 0 (corresponding to tolerance as zero).
2) Read the image to be compressed.
3) Process 'domains'

a. Scale the image by calculating the average values of each four-pixel group, then save the calculated values into an array ‘domain’

b. Divide the image (in ‘domain’) into overlapping domains (16x16 or 8x8)

c. Divide each domain block into 4 quadrants and calculate the variance of each quadrant.

d. Classify the domains into 24 classes (Figure 3.8) according to the order of the variances of the quadrants of the domain blocks. Record the position, the size and the class of the domain blocks in the corresponding class chain.

e. After processing the 16x16 domains, the procedure is repeated until you reach the smallest domains (4 x 4) as specified by the maximum partition exponent.

4) Record the parameters into the output file including image size, maximum partition, minimum partition exponents, and maximum and minimum values of the image.

5) Process ‘ranges’

a. Partition the original image into ‘ranges’ according to the 2DRE quadtree method. If the minimum partition is not reached: go ahead to continue the partition until the minimum partition is reached (e.g. 16x16 range).

b. Classify the range according to the same rule as did for domains, i.e. 24 classes based on the order of the variances of the quadrants of the block.

c. Search the domain class chains to find the domain which matches the current range best by calculating the RMSE between the domain and range as Y - AX + B (Y corresponds to the values in the range, X the values in the domain). Record the position and the rotating factor (if the domain has a different class than the ranges) of the domain with the smallest RMSE, and A and B.

d. Check a) if the smallest RMSE is less or equal to the tolerance and

b) if the maximum partition is reached.

i. If both are 'Yes', go to (5);

ii. If a) is 'Yes' and b) is 'No', go to (5);

iii. If a) is 'No' and b) is 'Yes', go to (5);

iv. If both are 'No', put a bit '1' as a flag into the output file, then

v. Divide the current range into 4 sub-ranges according to the quadtree method, then go to (2).

e. Write A, B, the rotating factor, the position of the domain with smallest RMSE, into the output compressed file.

6) Calculate the compression rate: the number of bytes of the original image divided by the number of bytes in the output compressed file.

The parameters for the encoding algorithm include:

- The rms tolerance threshold e_c
- The maximum depth of the quadtree partition;
- The minimum depth of the quadtree partition;
- The type of domain pool

- The maximum allowable scaling factor s_{max}
- The number of classes compared with a range.
- The maximum and minimum values of the image

An encoding of an image consists of the following data:

- The final quadtree partition of the image
- The scaling and offset values s_i and o_i for each range
- For each range, a domain that is mapped to it
- The symmetry operation (orientation) used to map the domain pixels onto the range pixels.

Figures 3.9 and 3.10 show an example of the partition and transformation; Table 3.1 shows the parameters saved in the compressed data file including the domains', ranges', and transformations'.

3.3.3 Decoding (Decompressing) Algorithm

Decoding an image consists of iterating W from any initial image. The quadtree partition is used to determine all the ranges in the image. For each range R_i, the domain $D_{i,}$ which maps to it is shrunk by two in each dimension by averaging non-overlapping groups of 2 x 2 pixels. The shrunken domain pixel values are then multiplied by si, added to oi, and placed in the location in the range determined by the orientation information. This constitutes a decoding iteration. The step is iterated until the fixed point is approximated, that is, until further iteration does not change the image or until the change is below some small threshold value.

The steps of the decoding (decompressing or restoring) procedure are as follows:

1) Determine the parameters of the restoring procedure:

a) Hypothetical image (the "initial image") name,

b) Compressed file name and

c) Number of iteration.

2) Read the input file (the compressed file).

3) Read the hypothetical image into an array 'image', which will be used as the domains. The hypothetical image can have any values (e.g. all 0's, or all 1's, etc.).

4) Get the parameters of the compressed image from the input file:

   a) Image size,

   b) Maximum partition,

   c) Minimum partition exponents, and

   d) Maximum and minimum values of the original image.

5) Prepare a blank image 'image1' for the restored image.

6) Read the transformations from the input file:

   a) Set 'level'=0.

   b) Checking level is less/more than minimum partition.

      i) If 'Yes', divide the blank image block into 4 sub-blocks, with level +1, put sub-blocks into a queue, then go to (3);

      ii) If 'No', i.e. level is equal to minimum partition, go to (4).

   c) Check the queue. If it is not empty:

      i) Obtain the first one and remove it from the queue,

      ii) Then go to (2); if it is empty, go to (7).

d) Checking if 'level' is less than maximum partition and the flag bit '1' in the input file.

i) If both are 'Yes', divide the current block into 4 sub-blocks

ii) With level +1, put them into the queue, then go to (3);

iii) If a) or b) or both are 'No', i.e. either the maximum partition is

iv) Reached or good transformation is encountered, then go to (4).

v) Get the transformation parameters from the input file:

A, B, rotating factor and position of the domain, then put them and the corresponding positions of the restored image into a transformation chain.

vi) Go to (3).

vii) Finish the procedure starting from step 6).

7) Do the transformations:

a) Set a counter=0;

b) heck if counter is greater than the predetermined iteration number.

If 'Yes', go to (8);

If 'No', go to (3).

c) Check the transformation chain.

If it is empty, counter+1, then go

To b); if it is not empty, go to d).

d) Get the parameters in the transformation chain, calculating the pixel values for the restored image by scaling the hypothetical image and using Y=AX+B, if necessary, rotating it (A, B and rotating factor are all in the transformation chain,

Y represents the restored value, and X the ‘initial value’ in hypothetical image), then put the pixel values into ‘image1’.

e) Go to c).

f) Copy ‘image1’ to ‘image’. The later will be the ‘hypothetical image’ or the domain of the next iteration.

g) Counter+1, then go to (2).

h) Finish the iteration.

8) Post-process the restored image by using maximum and minimum values of the original image.

9) Write ‘image’ into output file.

Figure 3.10 is the initial image of which all pixels are ‘white’, i.e. gray level is 255; 3.11 and 3.12 show the result of decompressing procedure at the first and the tenth iteration respectively. It can be seen that after ten iterations, the restored image became very similar to the original one.

For the past few years, there has been a tendency to combine different techniques and methods in order to obtain superior results in the field of image compression. In this study, this is also presented: fractal concepts, iterated function system, quadtree structure, and classical transformations are combined to acquire more efficient and effective methods of compressing and decompressing remotely sensed imagery.

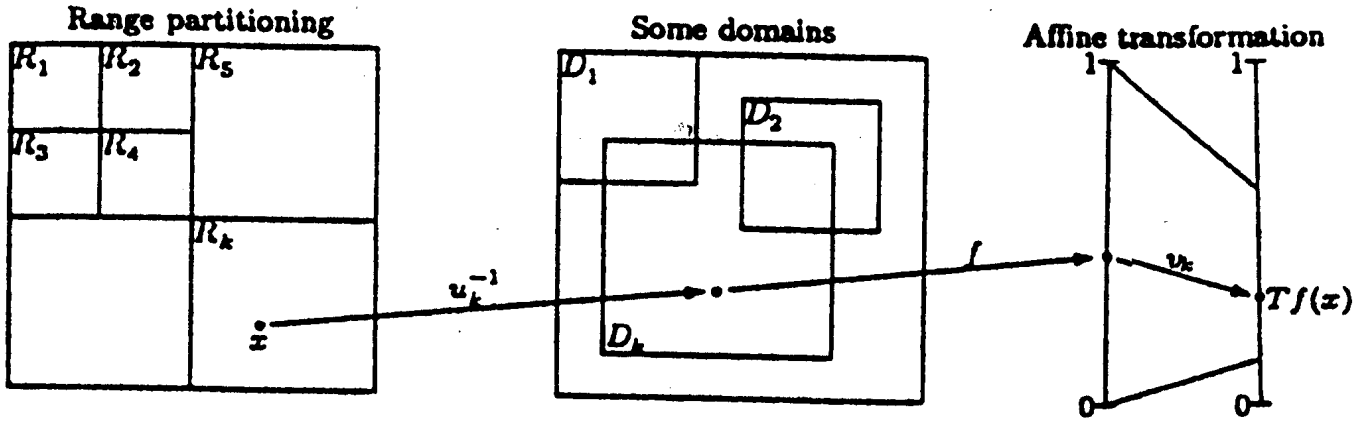


Figure 3.7 Partition with Self-Similarity
(Source: Saupe et al 1994)

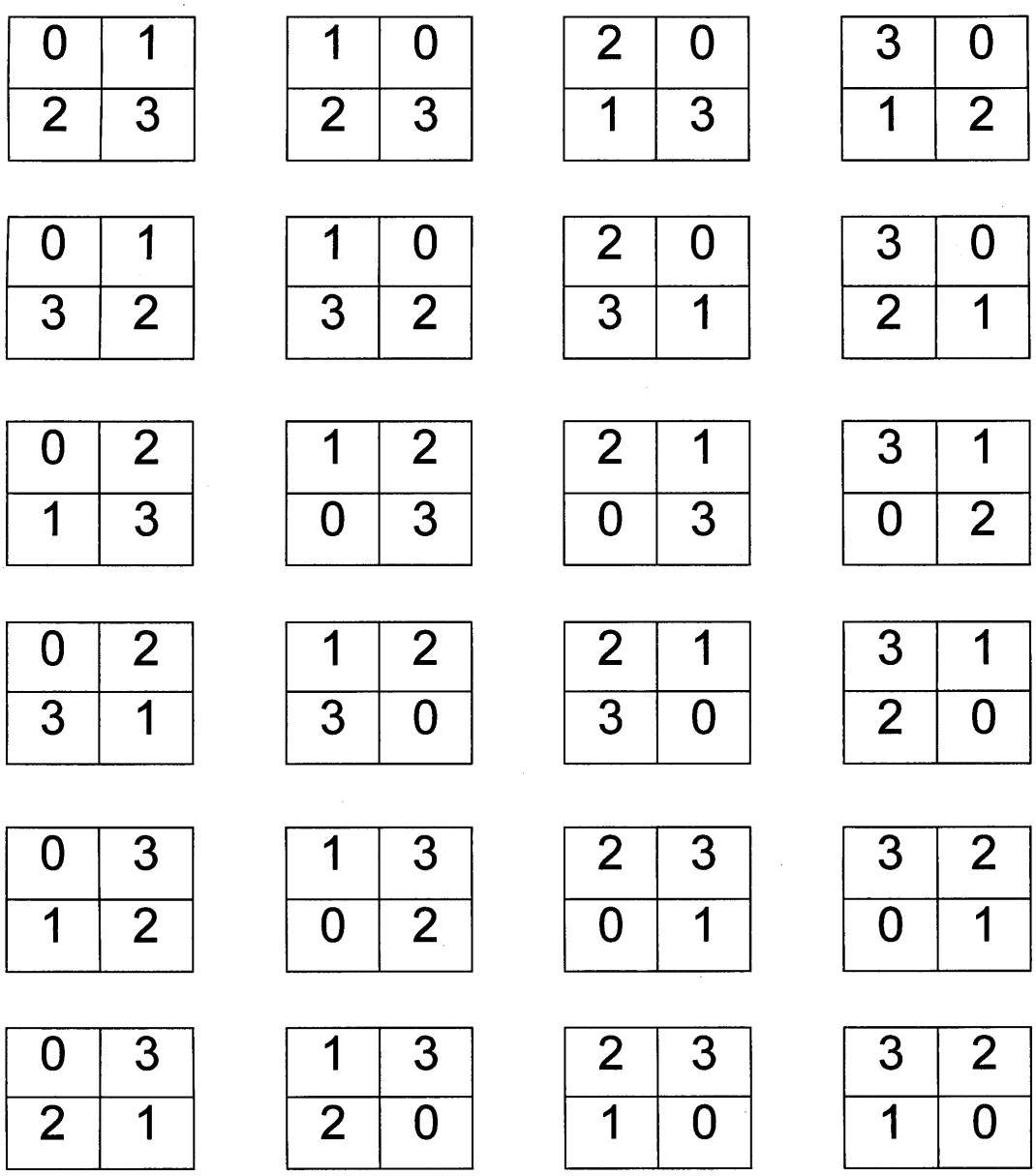


Figure 3.8 Twenty Four Classes
(0, 1, 2, and 3 represent the ordered values of variances of the quadrants.
There are twenty four situations for the whole combination)

Figure 3.9 Image of San Francisco

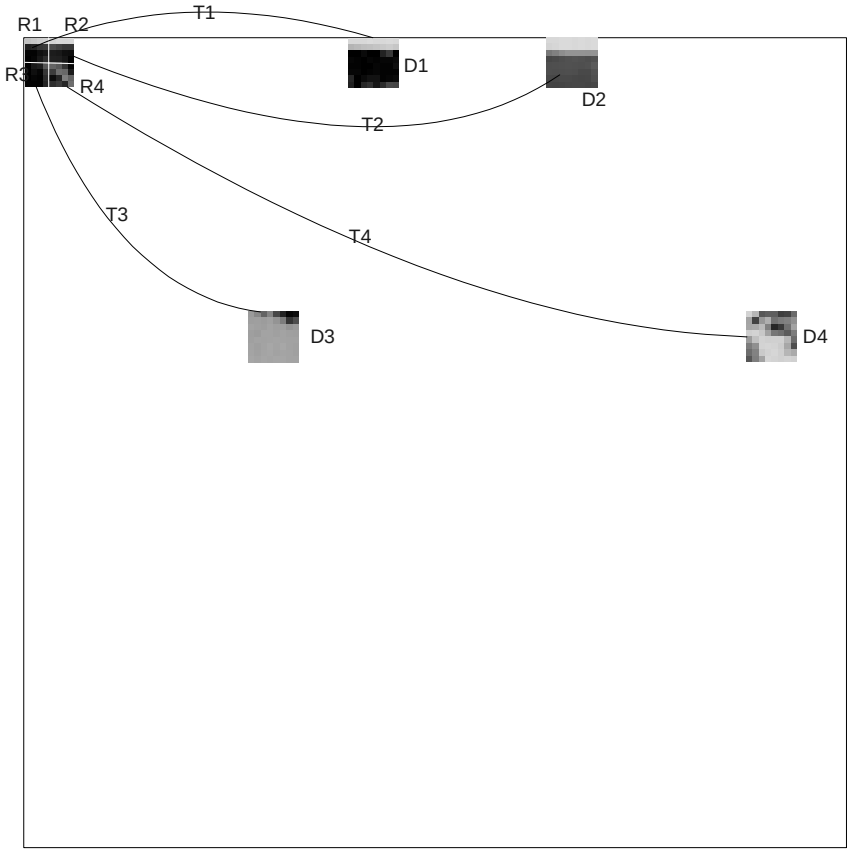


Figure 3.10 Domains, Ranges, and Transformation for the image in Fig.3.9
(R1 through R4: first four ranges in the image; D1 through D4: domains corresponding to R1 through R4; T1 through T4: transformations between Di and Ri, i=1,2,3,4)

Table 3.1 Parameters Saved in Compressed Data File

Value S	Value O	Symmetrical Operator	Domain Number
31	55	0	8
31	66	7	4
7	40	6	322
23	43	4	746

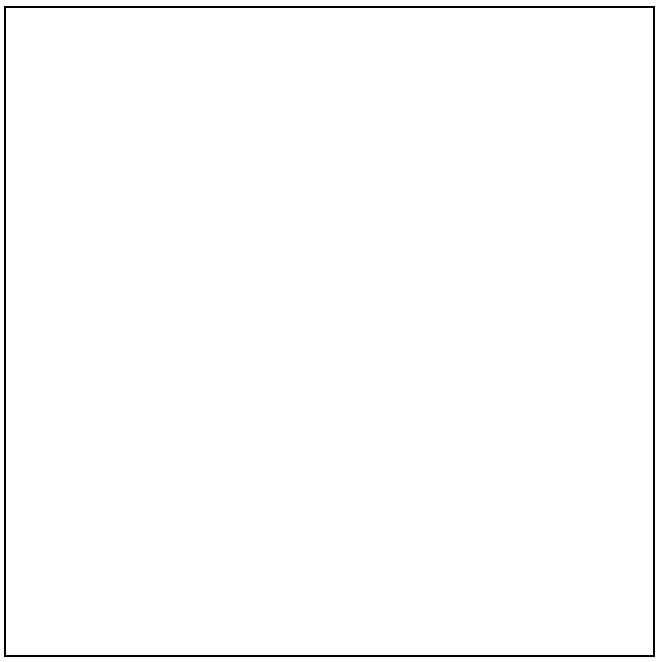


Figure 3.11 Initial image (all pixels' DN is 255)

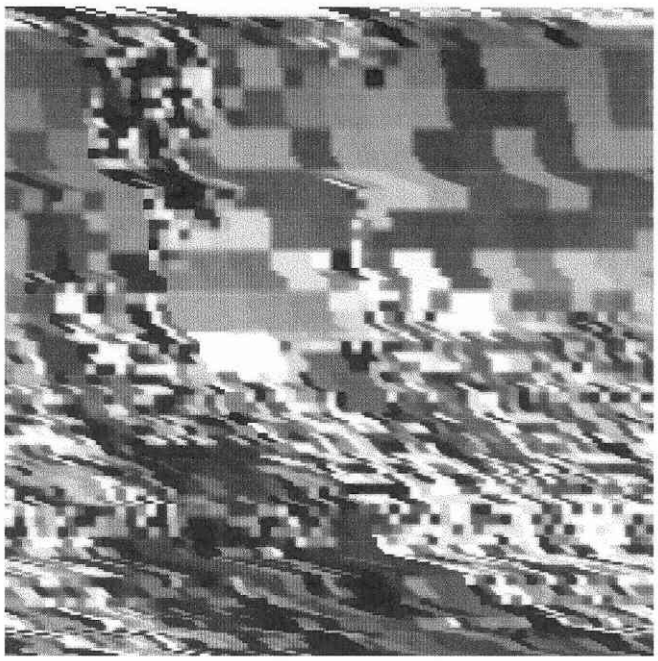


Figure 3.12 First Iteration of Decompression

Figure 3.13 Tenth Iteration of Decompression

CHAPTER 4

RESEARCH DESIGN

In this chapter, the research project will be described in detail, including: the study areas, data preparation, and experiment stages, including: algorithms testing on the remotely sensed images of various study areas; fractal dimension evaluating; and result analyzing.

4.1 Study Areas

From the research described in chapter 1 of this dissertation, the fractal dimension can be used primarily as an index for measuring the complexity of curves and surfaces. This observation leads to the idea that the fractal dimension can also be used to measure the complexity of geographical objects and landscapes, as well as the complexity of the corresponding remotely sensed images. The study areas must meet the requirement of testing the fractal compression algorithm. In other words, the study area must include different landscapes and geographical areas so that the fractal compression algorithm can be executed on the images and the results can be tested to see how the performance of the compression method changes with the different geographical areas of varying spatial complexities.

The State of Louisiana, which is included in the area between 33º North and the Gulf of Mexico in a north-south direction and between the Sabine River and Mississippi River, Pearl River, and Chandeleur Islands in a west-east direction, has diverse landscape and land cover types (Figure 4.1). In this area, there are different landscapes from place to place (Figure 4.2). Generally speaking, there are five main natural regions in the state, each of which has corresponding sub-regions in them (Kniffen 1988):

1) Coastal marsh,

   which includes fresh marsh and salt marsh vegetation, muck and peat soils, round and lagoon lakes;

2) Mississippi floodplain,

   has passes which have low elevation and relief, batture vegetation, mud-lumps and bars, distributary drainage, and inter-levee lakes;

3) Red River Valley,

   where there are drainage units, raft lakes, red soils, and bottom hardwoods;

4) Terraces,

   have bluff-lands with loessial soils, dendritic drainage, and bluff-land-woodland vegetation, flatwoods with mixed longleaf pine forest, bagols, pimple mounds, and flatwoods soils, and prairies with grassland, dendritic streams, and ice-age channels;

5) Hills.

   These regions are with maximum elevations and relief, dendritic and trellis drainage, interior salt domes, wolds or cuestas, and different forests.

It can be seen that the state of Louisiana enjoys diversity of landscape and natural regions with varying degrees of complexities. This meets the requirements of the research project. The study areas will be chosen from the state.

4.2 Data Preparation

Data preparation is based upon the criteria of complexity for choosing study areas: explicitly, the study areas should cover different landscapes so that the function of

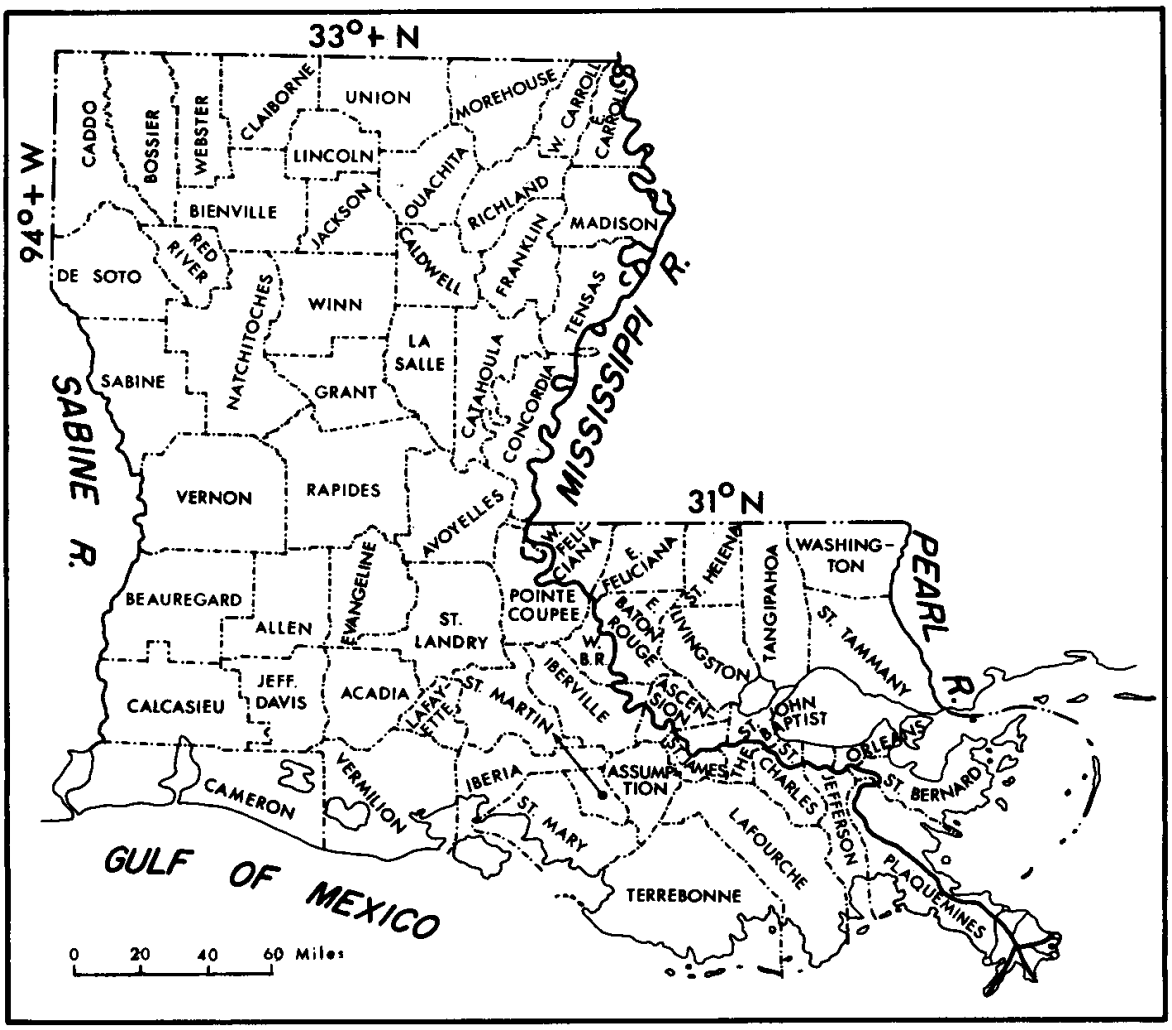


Figure 4.1 Map of Louisiana
(Source: Kniffen 1988)

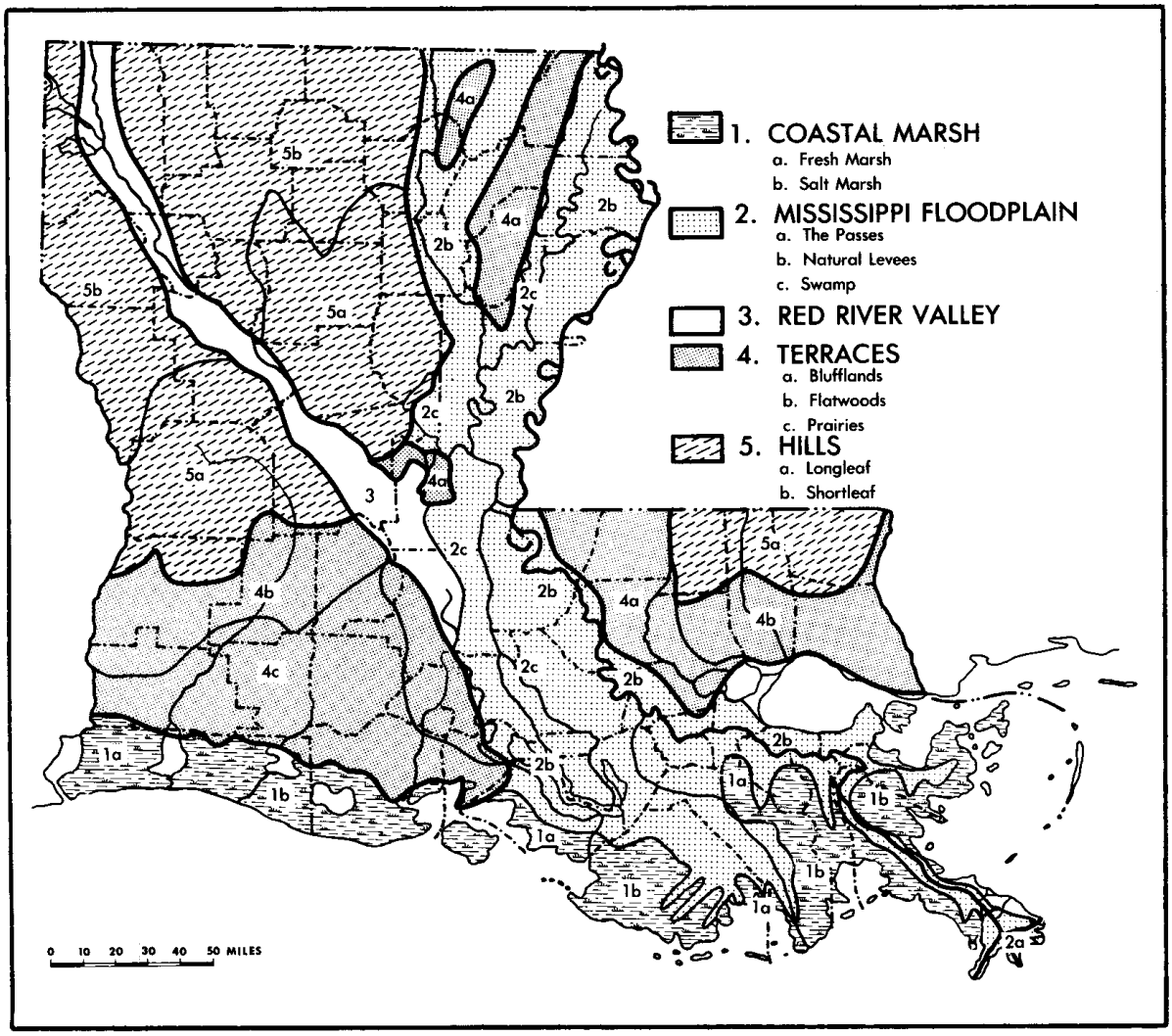


Figure 4.2 Natural regions of Louisiana
(Source: Kniffen 1988)

the fractal methods of measuring the complexities can be tested. The data ─ remotely sensed images -- were also chosen in accordance with the same considerations. In addition, different bands of the same image have different complexities, which also affect the compression rates of the fractal method. Subsequently, not only will the research provide the index of complexities of landscapes/land covers, but this research will also provide the measurement of spatial complexities of spectral bands.

In the most commonly used image data, Landsat data plays an important role in a variety of applications (Koutsias et al. 2000, Bailey et al. 2001, Beltran and Belmonte 2001, Carranza et al. 2002, Dhakal. et al 2002, Franklin et al. 2002, Wang et al. 2002). In the 1970s, Landsat I began an era of space-based resource data collection that changed the way science, industry, governments, and the general public view of the Earth. For the last 30 years, the Landsat program has successfully provided a continuous supply of synoptic, repetitive, and multispectral data of the Earth's land areas. The Landsat TM image data offers better spectral resolution than SPOT data: it has a good dynamic range in visual, NIR and MIR bands. Also the data offers better spatial resolution than MSS and AVHRR data: the corrected spatial resolution of 30 meters x 30 meters of Landsat TM images are better than MSS's 80 x 80 meters and AVHRR's 1 kilometer x 1 kilometer resolutions. As compared with other images, Landsat TM data are more widely used remote sensed images than others in a variety of different applications. The availability and the cost of the Landsat TM data are also reasonable. Therefore it is natural to choose Landsat TM images as the study data.

The Landsat TM images that are used in this research are all in the state of Louisiana. The images have six reflective bands including visible, NIR, and MIR with

spatial resolution of 30 meters and one thermal infrared band with coarser resolution of 120 meters. The images were acquired on November 29, 1992. The original image data were geo-rectified and resampled to 25 meters pixels for the six bands of visible, NIR, and MIR, and 100 meters pixels for the thermal infrared band by the Earth Observation Satellite Corporation (EOSAT). The projection of the image data set is Universal Transverse Mercator (UTM), zone 15, 1927 North American Datum (NAD 27). The image was provided by Mr. DeWitt Braud from the Department of Geography and Anthropology, Louisiana State University.

The eight study areas were selected from the Landsat TM scene. Each contains 512 x 512 pixels. The selected sub-images cover corresponding landscape types. Figure 4.3 shows the image of the whole state of Louisiana.

4.3 Experiment Stages

The experiment of the study includes several stages: study area chosen and data preparation, algorithms testing and implementing, results acquisition, as well as the analysis and discussion of the results.

4.3.1 Study Area Selection and Image Data Acquisition

For testing fractal as the measure of complexities of different geographical features and landscape/land cover types, eight different study areas were chosen:

Area 1: River ridge area;

Area 2: Agricultural area;

Area 3: Marsh area in South Louisiana;

Area 4: Water area (part of Lake Pontchartrain);

Area 5: Urban area in New Orleans;

Area 6: Swamp area;

Area 7 Forest area, and

Area 8: Mixed area with roads and vegetation.

Eight sub-images, corresponding to the eight study areas, were selected from the Landsat TM scene. All the sub-images contain seven separate bands: bands 1 through 3 are visible light ones. Band 1 is blue (wave length: 0.45-0.52 μm); band 2 is green (0.52-0.60 μm); band 3 red (0.63-0.69 μm); band 4 near-infrared (0.76-0.90 μm); band 5 near-infrared (1.55-1.75 μm); band 6 thermal-infrared (10.4-12.5 μm); and band 7 middle-infrared (2.8-2.35 μm). Each study area contains 512 x 512 pixels, covering area around 12.8 x 12.8 square kilometers. The selected sub-images cover different land cover types. Figures 4-4 through 4-11 display the false color composite images, which were contrast stretched from the red, green, and blue composite of bands 4, 5, and 3, for the eight study areas.

4.3.2 Algorithm Design and Implementation

First, the compression and restoration algorithms were applied on the images of the study areas (fifty six images in total: eight study areas by seven bands). Based on fractal geometry, the images were partitioned and encoded. The compression rate was calculated for the images after the compression is done. The processing time was also recorded. Then the restoration were carried out to get the reconstructed images. It was an iterative procedure in which the restored image would be obtained step by step. The result of every step was compared with the image acquired from the last step until the results of the last two steps were as close as possible. The statistics was also obtained and

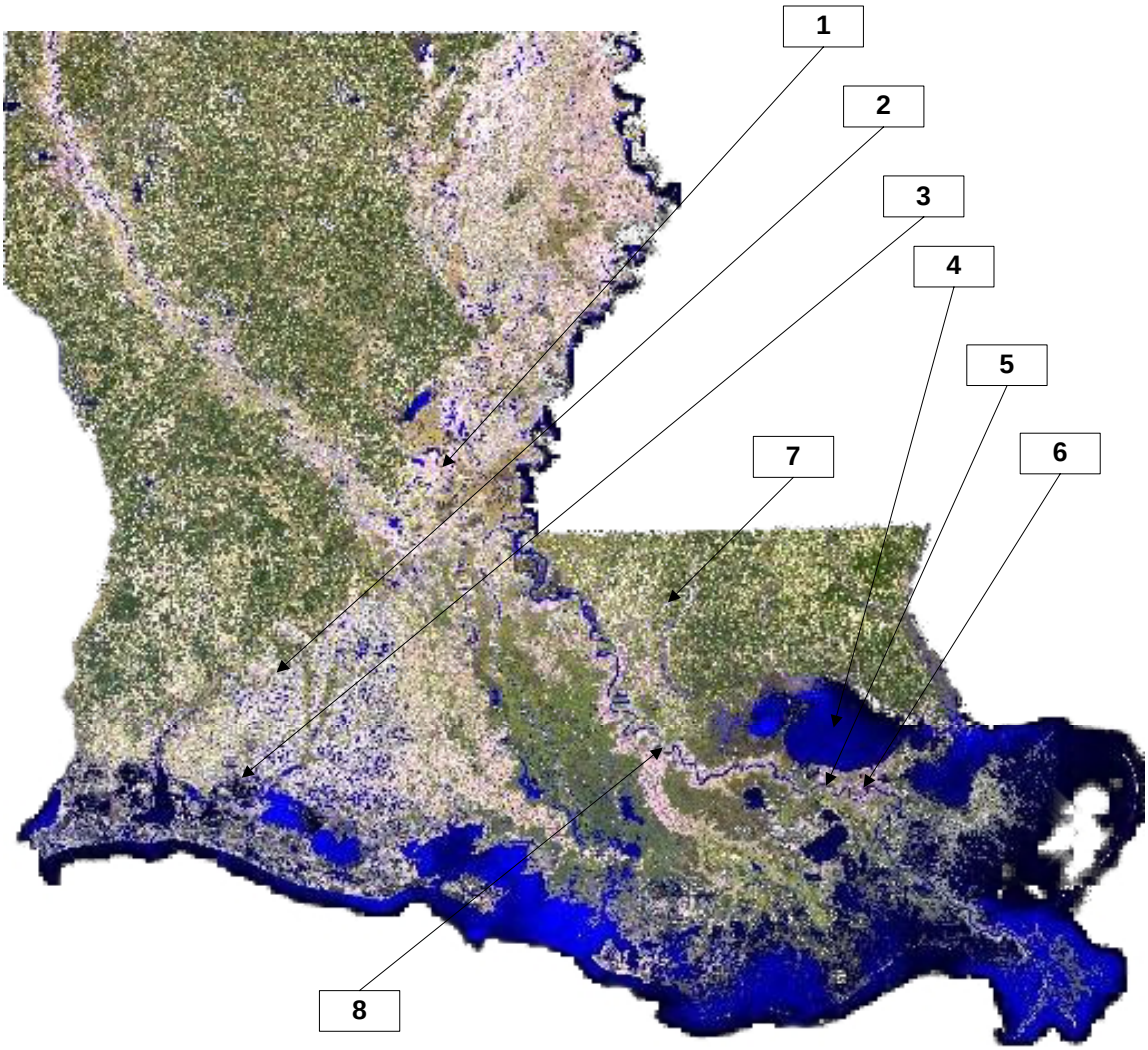


Figure 4.3 The Remote Sensing Image of Louisiana and Eight Study Areas

1. River ridge area
2. Agriculture area
3. Marsh area
4. Water area in Lake Pontchartrain
5. Urban area in New Orleans
6. Swamp area
7. Forest area

Mixed area with roads and vegetation the pixel profiles of the original and reconstructed images werecalculated for further analysis.

4.3.3 Analysis and hypothesis Testing

To evaluate the fractal image compression and decompression methods on the remote sensing images, the study first computes the performance measure, then evaluates the three research hypothesis listed in the introductory chapter.

1) Performance Measures

These measures will be computed and evaluated. They are:

i) Image fidelity

It is the most important criterion for evaluating the performance of the method. If the image fidelity is not substantial enough to fulfill pre-existing use criteria, the result will be meaningless. First, it can be tested visually. The classification method of the image will be used to test if useful information in the image is kept or lost. Quantitative comparison testing percentage of matching with the classification result of the original image will also be made to determine if the fidelity is preserved.

ii) Compression rate

Compression rate is the main standard for evaluating the compression algorithm since the goal of the procedure is to find an accurate and efficient method to solve the storage problem. The analysis also includes a comparison of the compression rates for different land covers, such as vegetation, water body, water/land boundary, and urban area to test how different landscapes affect the rate.

iii) Time complexity of the encoding/decoding procedure

The computational efficiency is essential for image compression and decompression. If the encoding/decoding procedure is too time-consuming, the value of the algorithm, or the improvement of the compression rate may not be meaningful because it may not be practical or operational.

2) Research Hypothesis Test

i) Hypothesis I: Fractal compression methods perform better than JPEG and WinZip;

ii) Hypothesis II: The higher the spatial complexity, the lower the compression rate;

iii) Hypothesis III: Changing pixel resolution will change the compression rate.

Different types of land cover may result in different compression rates in the corresponding areas on the remotely sensed imagery. Some types of land cover, appearing as homogeneous areas, may have higher compression rates, while some heterogeneous areas may have lower compression rates. Images with higher fractal dimension, have higher spatial complexity, and the compression rates in these areas are expected to be higher than those with lower fractal dimension. Therefore, to test these hypotheses, the fractal dimension will be computed for all bands in all images using the software called ICAMS (Image Characterization and Modeling System) (Lam et al. 1998). The comparison between different types of land cover will also be analyzed.

In the next chapter, the results of the compression and decompression will be displayed, explained, and analyzed. The results will include visual comparison of original and reconstructed images, including:

2) Profiles of the images,
3) Gray level distributions for the different images between the original and the reconstructed image,
4) Statistics of the images,
5) Compression rates and processing time,
6) The fractal dimensions of the images and the relationship between land cover type and compression rates.

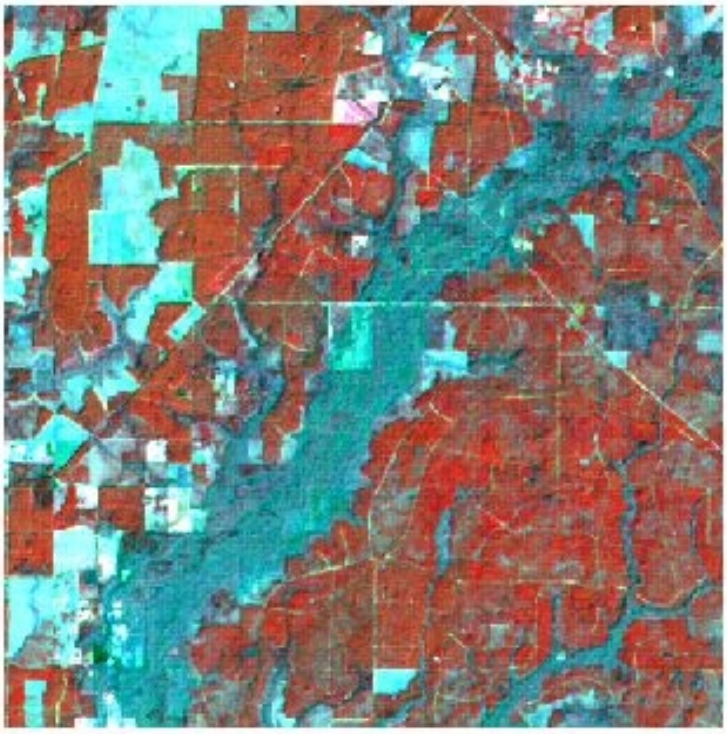


Figure 4.4 Study Area 1: River ridge area

Figure 4.5 Study Area 2: Agricultural area

Figure 4.6 Study Area 3: Marsh area in South Louisiana

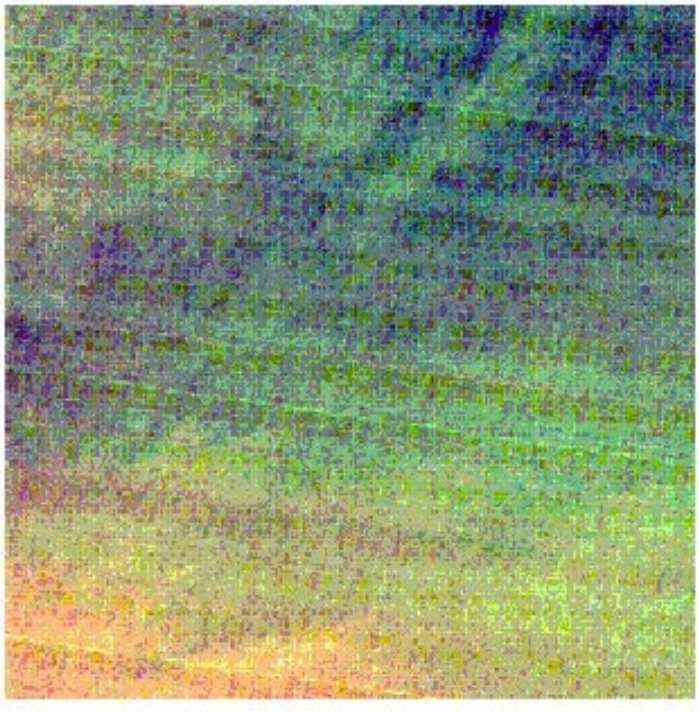


Figure 4.7 Study Area 4: Water area (part of Lake Pontchartrian)

Figure 4.8 Study Area 5: Urban area in New Orleans

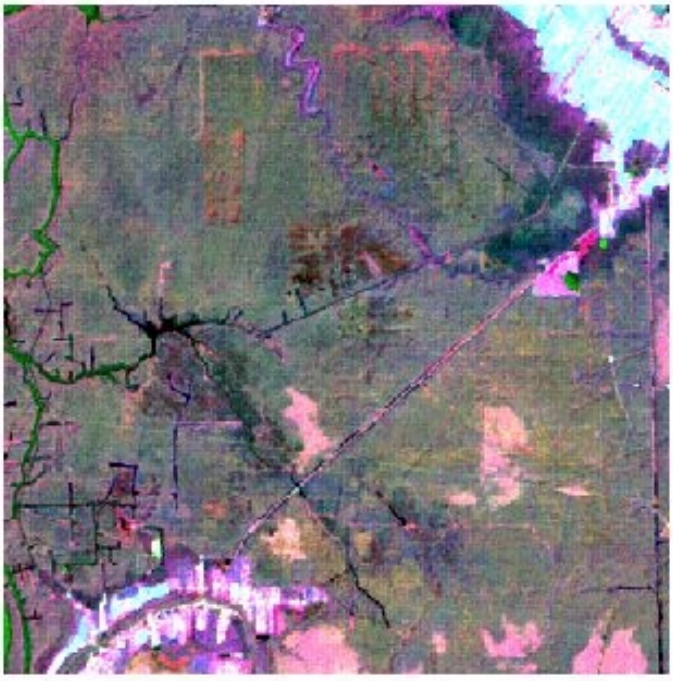


Figure 4.9 Study Area 6: Swamp area in South Louisiana

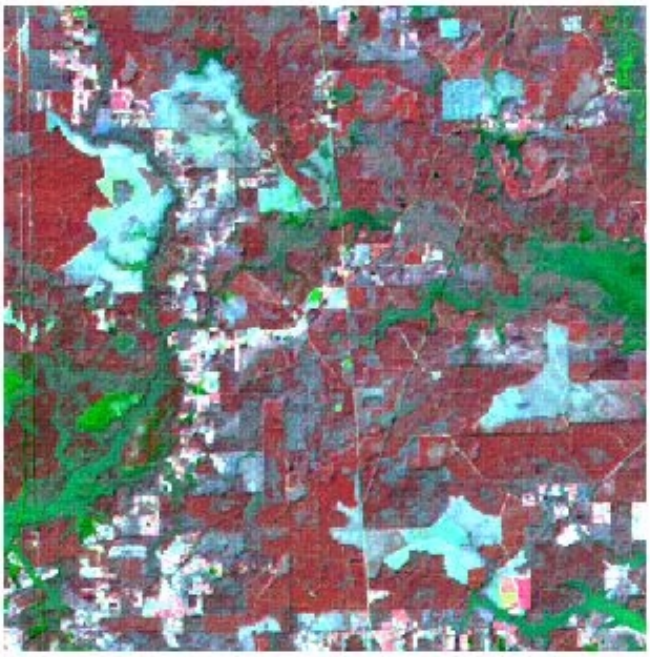


Figure 4.10 Study Area 7: Forest area

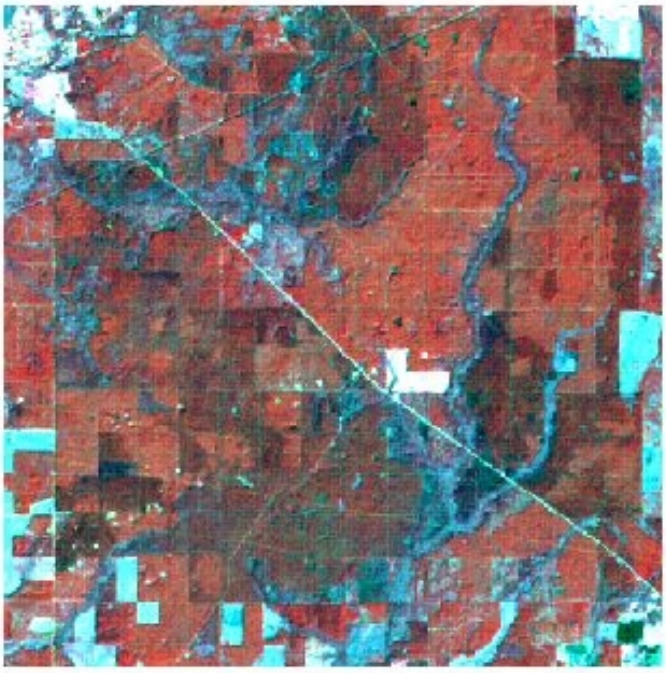


Figure 4.11 Study Area 8: Mixed area with roads and vegetation

CHAPTER 5

RESULTS AND DISCUSSION

In this chapter, results from applying the fractal compression and decompression algorithms described in chapter three to the eight study areas are reported. The results include the comparisons of the original and restored images; fidelity evaluation; analysis of compression rates on different land covers and different resolutions; and discussion on the efficiency of the algorithms, as well as a comprehensive evaluation of the research method in detail. Three research hypotheses were also tested. For testing the research hypotheses 1, the results of fractal compression method are compared with the other two commonly used data compression methods, JPEG and WinZip. For the hypothesis 2, the effects of spatial complexity to the compression result is also tested; besides the implementation of the algorithms, a series of image processing and statistical methods, as well as the software ICAMS (Quattrochi et al. 1997; Lam et al. 1998), an application module measuring the fractal dimensions of each image in the study areas, was also used to analyze the complexity of the data and the relationship between the compression results and the complexity. Finally, for hypotheses 3, the effects of pixel resolution on the compression results are analyzed.

5.1 Results of Fractal Image Compression

The images of the eight study areas were subset by band as separate images. Each sub-image contains one band. The fractal compression algorithm subsequently was implemented on each sub-image to obtain the compressed data file for each image file. The compression rates were calculated when compression was finished. Afterwards, the decompression algorithm was applied to each compressed data file to reconstruct the

images. The comparisons between the original and restored images for two study areas are shown in Figure 5.1 through Figure 5.14 in this chapter, and for the other study areas as Figures A.1 through A.42 in Appendix A. In these figures, part (a) is original image and part (b) is the restored image, respectively. From visual comparisons, it can be seen that the original and restored images appear similar to each other.

Besides the visual comparison of the original and restored images, the profiles of all images (both the original and restored) as well as the distributions of the difference images between the original images and their restored counterparts are obtained and displayed in Figure5-15 through Figure 5.28 in this chapter and Figures B.1 through B.42 in Appendix B. From the profiles (the distribution of gray levels) of the images, it can be seen that the original and reconstructed images match each other almost exactly. However, some are with slight differences. For example, in Figure 5.15 (b), which is the gray level distribution of the difference image (showing the arithmetic difference between the loaded reference image and a previously calculated potential-field image) of band 1 between the original and reconstructed images of study area 4. The maximum differences of gray levels are 4 while most others are 2 and 1. Others are even better: most of the difference images have the maximnu different gray level below 2. Some are only 0. An issue that should be emphasized is: not only are the differences between the original and reconstructed images at every pixel very small but also the distributions of difference are very similar. The difference images of study area-4 display the prevalent difference: which is 4, as shown in Figures B.1 (b), B.3 (b), B.4 (b), and B.6 (b). The best ones are shown in figures 4.1 (b), B.11 (b), B.12 (b), and B.20 (b), with small difference: either 1 or 0. More statistical results are listed in Tables 5.1 and 5.2, which

display the minimum values, maximum values, means, and standard deviations of each image. These descriptive statistics show that the original and the restored images are very similar.

An image classification method is also applied to make further comparisons between the original and restored images. In this part of the study, unsupervised classification is used to get 10 classes (study area 4 only has seven classes) for the composite images, which combine the six bands of visible, NIR and MIR images of each study area (both original and restored). Tables 5.3 and 5.4 show the results of both original and restored images.

Compression rates are calculated at the end of each processing of the compression algorithm on the original images. The size of a 512 x 512 image is divided by the size of the compressed data file (in bytes). The processing times of both compression (encoding) and decompression (decoding) are also recorded. Tables 5.5 and 5.6 display the compression rates of each image, with compression and decompression times.

5.2 Evaluation of Fractal Image Compression

Based on the results shown in the previous sections, it is apparent that the algorithms of fractal compression and decompression developed and implemented in this study are efficient, effective and consequently helpful methods for solving storage problems and providing an approach to reveal the relationship between the complexities of geographical features and the compression rates. Below is a discussion of each criterion.

5.2.1 Fidelity Evaluation

Figures 5-1 through 5-14 and Figures A-1 through A-42 provide the direct visual comparison of both the original and the restored images. Figures 5-15 through 5-28 and Figures B-1 through B-42 show applicable graphs for the purpose of displaying greater detailed information about the difference of the two kinds of images. From the part (a) of each graph, it can be seen that the distribution of gray levels of the original and restored image matches each other very well, almost exactly. That can be seen from the part (b) of the graphs and the statistics of the images shown in Tables 5.1 and 5.2. The largest difference between all original and restored images is 4, and appears only once. Others are all below this value. Most of the difference of maximum values range fall between 0 and 1. The minimum and mean values are even better. Furthermore, the parts (b) show that overall differences between the two kinds of images are also small. The values of the differences are mostly located in the regions between zero and one, and rarely reach 2 or 3. Consequently, the profiles reveal not only are the differences between the original and reconstructed images at every pixel very small but also the distributions of difference are very even.

More evidence is provided by the result of the unsupervised classification applied on both original and restored images. Every composite image consisting of all but the thermal infrared band is classified into 10 classes (the only exception is that the image of study area 4: has only 7 classes) and the results of each pair of the original and restored images for the same study area are put together to make comparison. Then the degree of matching can be clearly tested. Tables 5.3 and 5.4 show the numbers of pixels in each class for both original and restored images corresponding to the study areas and the

percentages of matching between the two kinds of images. Again, the results show that the restored images match the original ones faithfully. All of the classes have better than 95% matching (in fidelity matching most of them reach 99% and above while in the over-matching, the largest mismatching value is 3.5%. Most mismatching are below the 1% threshold).

Based on the results and the analysis above, it can be said that the restored images maintain the quality of the original images within a very acceptable range of fidelity. The conformity criterion of the evaluation of the compression and decompression algorithms is met.

5.2.2 Efficiency Evaluation

This evaluation includes two categories: compression rate and processing time. First, the compression rate will be evaluated. Over all study areas and all bands, the lowest compression rates are in the order of five (Table 5.6). The maximum compression rates occur in images of study area 4, which is a water area. All are higher than 400. This phenomenon can be explained due to the smoothness of water area in all the bands. Similarly, the images of band 6 of all the study areas have high compression rates, and the average is 319.33. This is because the thermal infrared band 6 has lower pixel resolution. Accordingly, the images of this band appear smoother than other bands. Table 5.10 lists the average compression rates of separate areas. Area 4 has the highest average compression rate while area 5 has the lowest. Table 5.11 shows the average compression rates of separate band, where band 2 has the highest, and bands 4 and 5 have the lowest values (band 6 is not considered).

Table 5.1 Statistics of the Images of Study areas 1 to 4

AREA	BAND	ORIGINAL					RESTORED				
		min	max	mean	std	c.v.	min	max	mean	std	c.v.
5	1	38	219	54	9.864	0.183	39	220	55	9.913	0.183
	2	9	145	22	6.073	0.276	10	143	23	5.942	0.276
	3	8	182	23	8.272	0.360	9	185	24	8.289	0.360
	4	4	163	33	10.621	0.322	3	160	33	10.022	0.322
	5	1	227	43	15.724	0.366	0	225	42	15.127	0.366
	6	78	119	110	3.639	0.033	78	118	109	3.685	0.033
	7	1	150	21	8.81	0.420	0	148	20	8.688	0.420
6	1	29	111	50	2.179	0.044	30	109	49	2.099	0.044
	2	11	48	19	1.729	0.091	12	46	19	1.685	0.091
	3	9	57	19	2.342	0.123	10	54	18	2.331	0.123
	4	5	84	30	6.317	0.211	3	82	30	6.298	0.211
	5	1	148	34	8.83	0.260	0	146	35	8.63	0.260
	6	103	113	107	1.099	0.010	102	113	107	1.241	0.010
	7	1	143	15	4.146	0.276	0	143	14	4.132	0.276
7	1	29	97	45	2.739	0.061	30	97	44	2.74	0.061
	2	11	50	16	1.899	0.119	12	49	16	1.809	0.119
	3	6	138	15	3.377	0.225	8	136	15	3.357	0.225
	4	4	142	32	6.042	0.189	2	141	31	5.977	0.189
	5	2	93	32	11.492	0.359	0	92	31	11.212	0.359
	6	70	110	103	1.725	0.017	70	111	103	1.794	0.017
	7	1	46	12	5.386	0.449	1	48	12	5.349	0.449
8	1	35	84	46	2.626	0.057	37	86	46	2.485	0.057
	2	9	140	16	1.733	0.108	10	140	16	1.756	0.108
	3	5	108	15	3.147	0.210	6	107	15	3.046	0.210
	4	12	164	37	4.65	0.126	13	153	36	4.508	0.126
	5	10	148	28	11.508	0.411	12	147	29	11.279	0.411
	6	110	124	115	1.405	0.012	110	124	114	1.568	0.012
	7	2	62	9	5.55	0.617	3	63	10	5.448	0.617

Table 5.2 Statistics of the Images of Study areas 5 to 8

AREA	BAND	ORIGINAL					RESTORED				
		min	max	mean	std	c.v.	min	max	mean	std	c.v.
5	1	38	219	54	9.864	0.183	39	220	55	9.913	0.183
	2	9	145	22	6.073	0.276	10	143	23	5.942	0.276
	3	8	182	23	8.272	0.360	9	185	24	8.289	0.360
	4	4	163	33	10.621	0.322	3	160	33	10.022	0.322
	5	1	227	43	15.724	0.366	0	225	42	15.127	0.366
	6	78	119	110	3.639	0.033	78	118	109	3.685	0.033
	7	1	150	21	8.81	0.420	0	148	20	8.688	0.420
6	1	29	111	50	2.179	0.044	30	109	49	2.099	0.044
	2	11	48	19	1.729	0.091	12	46	19	1.685	0.091
	3	9	57	19	2.342	0.123	10	54	18	2.331	0.123
	4	5	84	30	6.317	0.211	3	82	30	6.298	0.211
	5	1	148	34	8.83	0.260	0	146	35	8.63	0.260
	6	103	113	107	1.099	0.010	102	113	107	1.241	0.010
	7	1	143	15	4.146	0.276	0	143	14	4.132	0.276
7	1	29	97	45	2.739	0.061	30	97	44	2.74	0.061
	2	11	50	16	1.899	0.119	12	49	16	1.809	0.119
	3	6	138	15	3.377	0.225	8	136	15	3.357	0.225
	4	4	142	32	6.042	0.189	2	141	31	5.977	0.189
	5	2	93	32	11.492	0.359	0	92	31	11.212	0.359
	6	70	110	103	1.725	0.017	70	111	103	1.794	0.017
	7	1	46	12	5.386	0.449	1	48	12	5.349	0.449
8	1	35	84	46	2.626	0.057	37	86	46	2.485	0.057
	2	9	140	16	1.733	0.108	10	140	16	1.756	0.108
	3	5	108	15	3.147	0.210	6	107	15	3.046	0.210
	4	12	164	37	4.65	0.126	13	153	36	4.508	0.126
	5	10	148	28	11.508	0.411	12	147	29	11.279	0.411
	6	110	124	115	1.405	0.012	110	124	114	1.568	0.012
	7	2	62	9	5.55	0.617	3	63	10	5.448	0.617

The results show that different bands have different compression rates while different land cover types also have different rates. Study area 4, the water area is a distinctive case. It has the highest compression rate because area 4 possesses the characteristics of both smoothness and homogeneousness. Other land cover types also have relatively high compression rates except study area 5. Study area 5 is an urban area in New Orleans, Louisiana. Its compression rate is the lowest because of its complex geographical features and structures, but it is still greater than 5. When compared to traditional methods, the maximum is below 4 (Chen et al. 1987). As shown in the next section, the most commonly used non-fractal compression methods, JPEG and Winzip, were tested in this research, and the fractal methods obtained much higher compression rates in most cases.

The computational time of fractal compression and decompression is also shown in Tables 5.7 and 5.8. It can be seen that almost all the compressing procedures are shorter than one minute. Moreover, more than half of the compressing procedures are done immediately. Again, study area 5, an urban area in New Orleans has the longest average compression time because of its complex geographical features and textures, while study area 4, the water area has the shortest compression time due to both its smoothness and homogeneousness. All decompressing times are shorter than 30 seconds and all are similar in time, thus suggesting a uniform method of compression. This is because the decompression procedures take the same numbers of iteration.

In summary, in terms of image fidelity, compression rate, and computational time, the fractal algorithms perform very well. The algorithms reach higher compression rates

compared to traditional methods based on the results reported from the literature, and the processing time is also shorter.

5.3 Hypothesis I: Comparison with Different Compression Methods

To test the first research hypothesis of this study, i.e. fractal image compression methods are more accurate and efficient than other commonly used compression schemes, two non-fractal compression methods, JPEG and Winzip, were also tested on the same data set as the fractal compression techniques. Table 5.5 compares the compression rates using the fractal, Winzip, and JPEG compression techniques.

It can be seen that for every band of images in all areas, the fractal method has the highest compression rate. In the most complex study area, the urban area in New Orleans (study area 5), all of the compression methods have the lowest rates as compared with themselves and other areas. The fractal still has the highest rate, which is 2 to 3 times higher than the other two. JPEG, on the other hand, has the lowest compression rate, even lower than 1 for study areas 3 and 5. This means that the size of a compressed file is larger than the original. In the smoothest area, the water in Lake Pontchartrain (study area 4), the fractal method has a much higher compression rate than the other two.

When the convenience of the usage of the three methods was also considered, only JPEG compressed images can be displayed directly, because the compressed data are still images. On the other hand, the results of fractal and Winzip compression cannot be displayed under these conditions because they maintain different formats. Instead, they can only be displayed after decompressing or unzipping. However, all the results, i.e. the compressed data files of the three methods, cannot be directly applied or processed. This

includes but is not exclusive of classification, information extraction, and other processing.

From the above comparisons, the fractal method is shown to perform better than the other two methods: JPEG and WinZip.

5.4 Hypothesis II: Effects of Spatial Complexity

The second hypothesis is that the spatial complexity will affect the compression rate. The more complex the image, the lower the compression rate; the less complex the image, the higher the compression rate. In other words, if smoother or more homogeneous images have lower spatial complexity, then higher compression rates can be expected, while coarser or more heterogeneous images have higher complexity and lower compression rates. These principles were tested in this study. Table 5.7 lists the relationship between compression rate and complexity. The measurements of complexity in images include fractal dimension, and coefficient of variation (c.v.), which is the standard deviation divided by the mean. Fractal dimension is a measure of spatial complexity; whereas c.v. is a measure of non-spatial complexity.

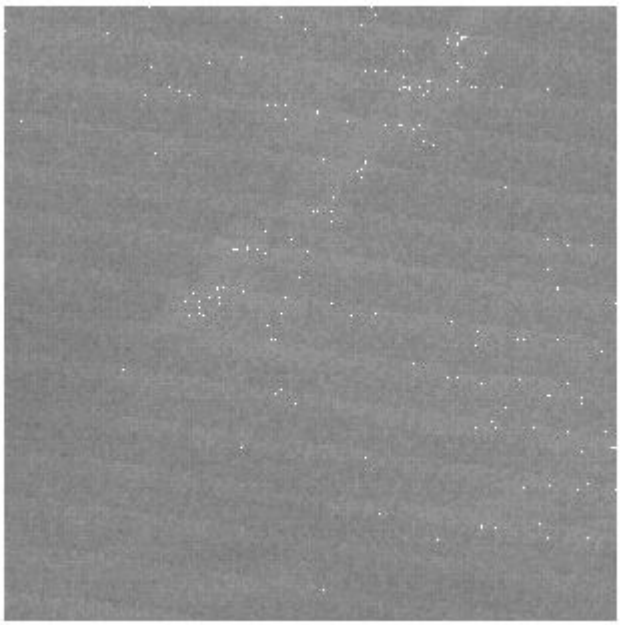
It can be found that the smoothest area, the water in Lake Pontchartrain (area 4), has less complexity for both the spatial and the non-spatial measure, and the compression rates for the images of all bands are the lowest in the images. In the most complex area, the urban areas in the City of New Orleans, the situation is the opposite: the indicators of complexity: fractal dimensions, and c.v., are high and compression rates are the lowest among all images.

5.5 Hypothesis III: Effects of Resolution

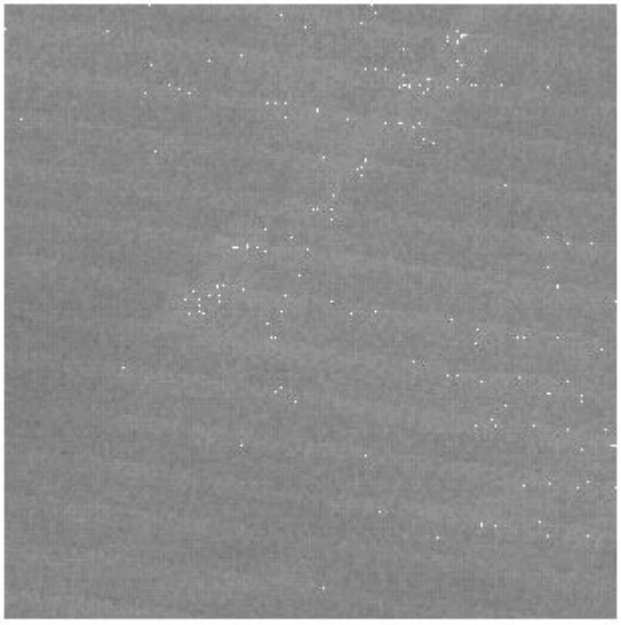
The third hypothesis is: Changing scale or resolution of images will result in a change in the fractal compression rate. More specifically, it is expected that increase in pixel size (i.e. decrease in spatial resolution) will lead to a higher compression rate. In this research, two study areas, the urban area in New Orleans (Study Area 5), which has the most complex geographical structures and features, and water in Lake Pontchartrain, which is the smoothest area (Study Area 4), are tested. The resolutions of the images of all the seven bands in the two areas were resampled by using 3 x 3 window to average the values. The resolution was changed from 30 meters to 90 meters (band 6 from 120 meters to 360 meters) so that they became smoother than the original images. Table 5.9 shows the results.

The results show that the complexities of both areas become lower when the resolution of the images is lower: fractal dimensions become lower while fractal compression rates get higher for all bands of both study areas. The increase in compression rates for the urban area is more than the increase of compression rate for the water area. The former is about 3 to 4 times higher than the original while the latter is smaller. The reason is because the water area is already very smooth, the effect of resampling or smoothing for the water will be smaller than for the urban area. This is consistent with the fact that compression rate of band 6 image of water area is not higher than the other bands as the complexity of band 6 of water area is already very low.

The effect of resolution is consistent with the second hypothesis: changing resolution results in a change of complexity; and changing complexity will result in a change of compression rate.

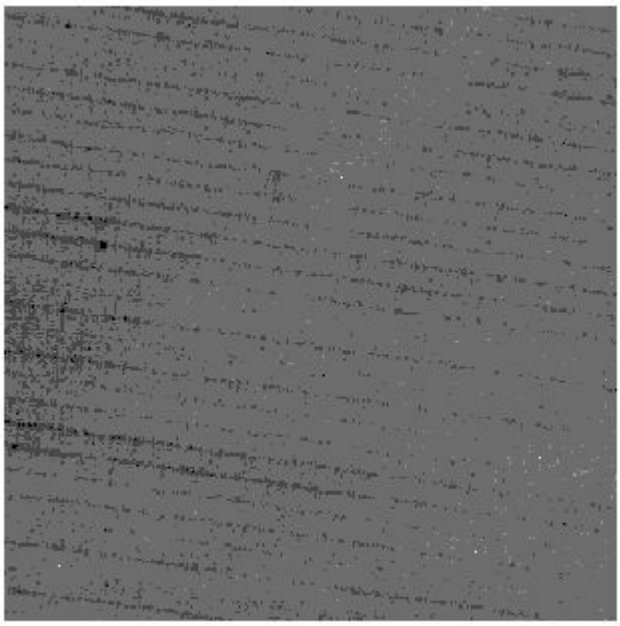


(a)

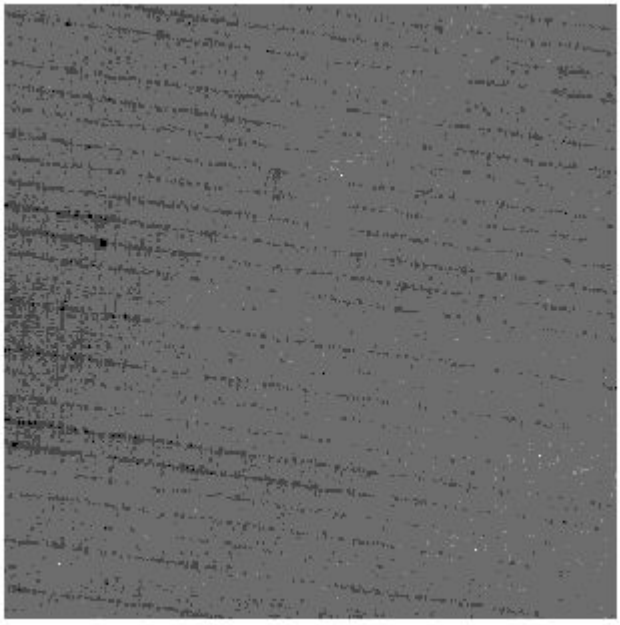


(b)

Figure 5.1 Original and Restored images of Area-4 (water), Band 1

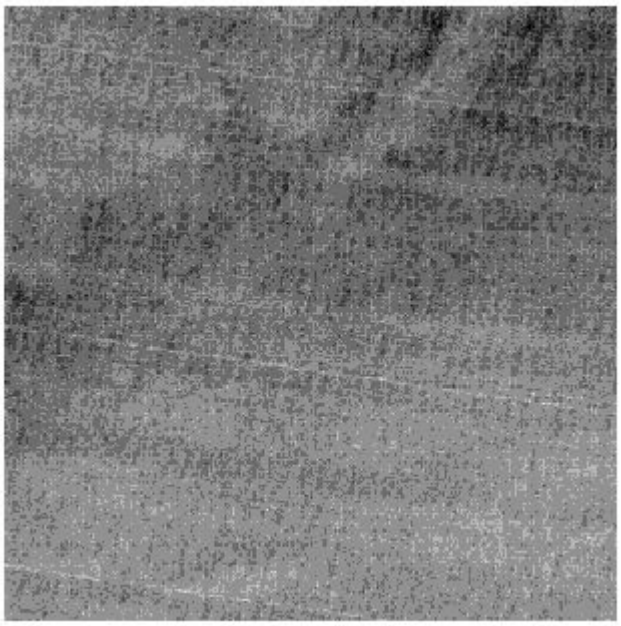


(a)

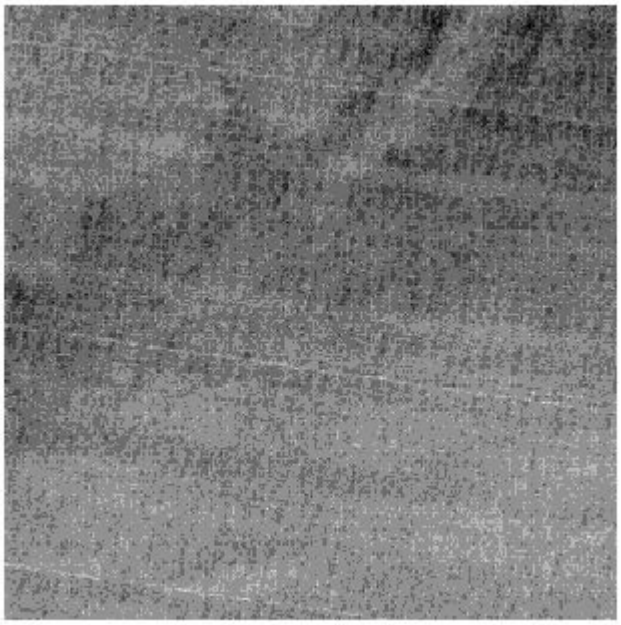


(b)

Figure 5.2 Original and Restored images of Area-4 (water), Band-2

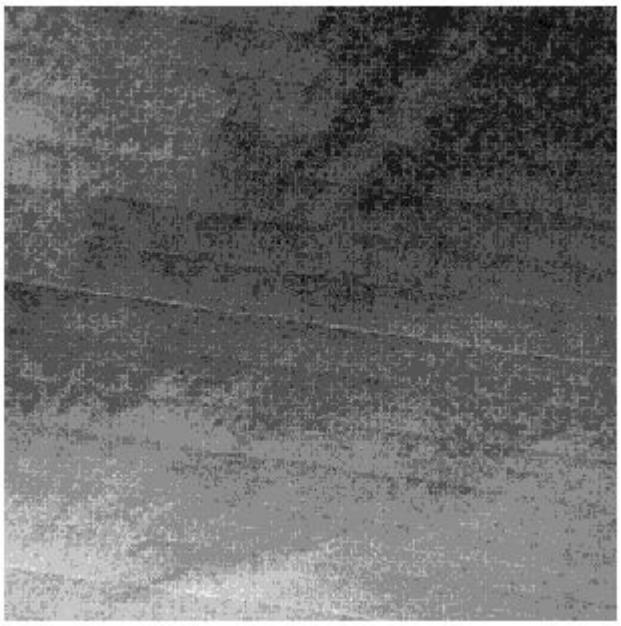


(a)

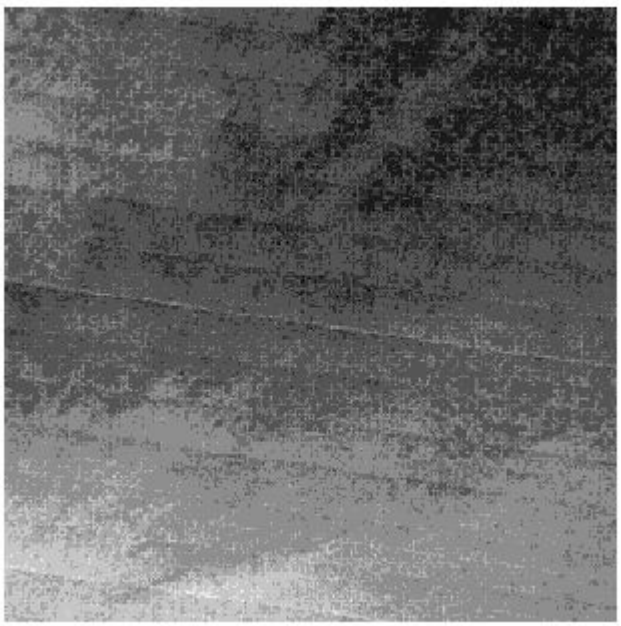


(b)

Figure 5.3 Original and Restored images of Area-4 (water), Band-3

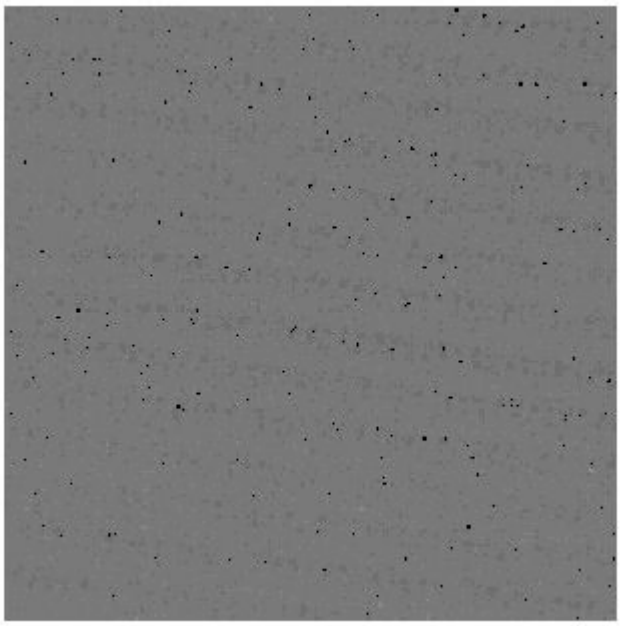


(a)

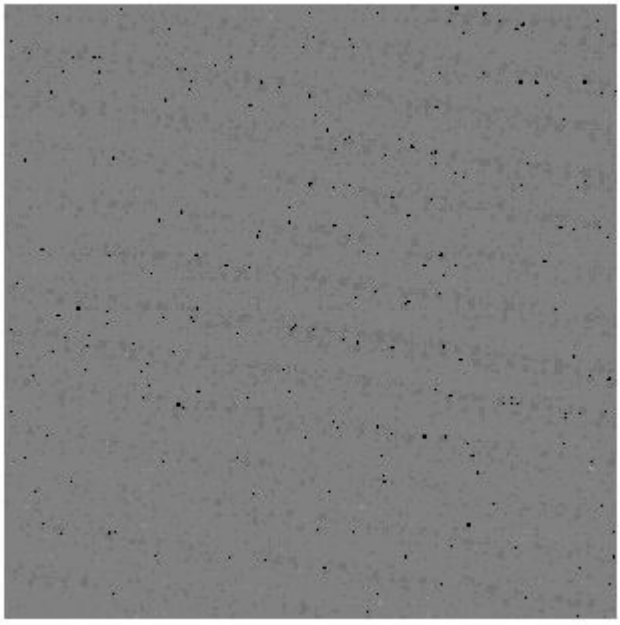


(b)

Figure 5.4 Original and Restored images of Area-4 (water), Band-4

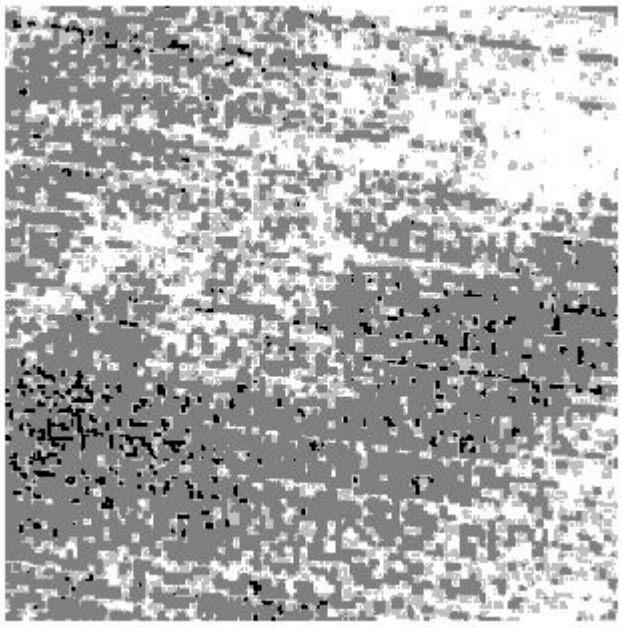


(a)

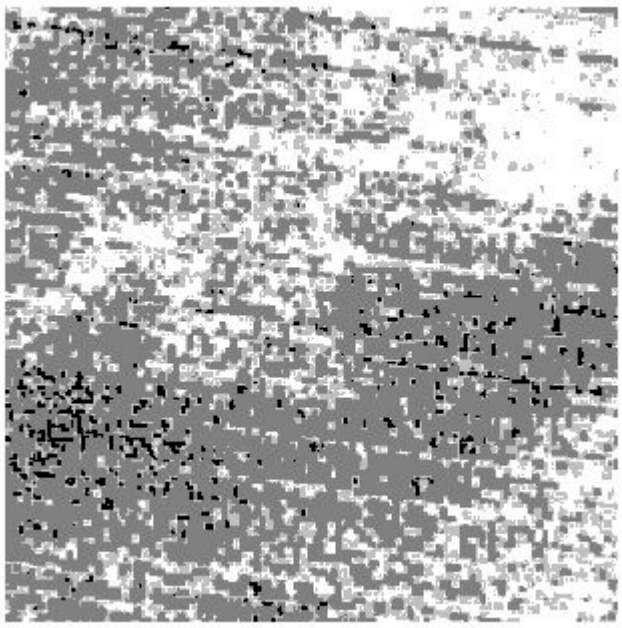


(b)

Figure 5.5 Original and Restored images of Area-4 (water), Band-5

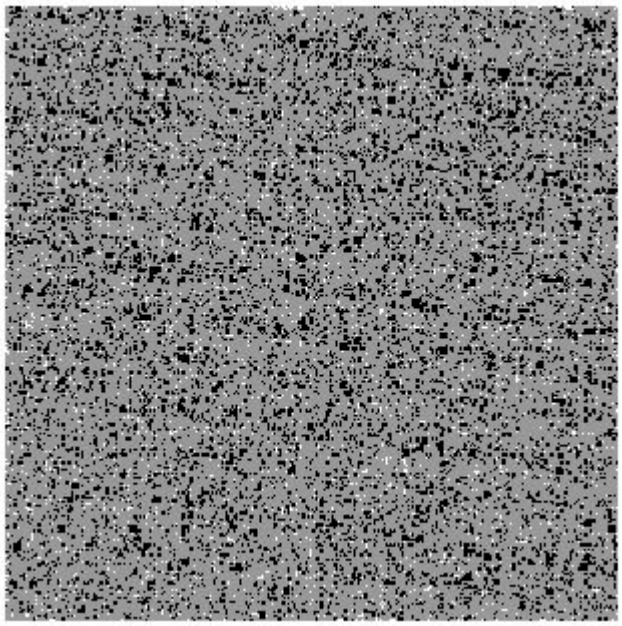


(a)

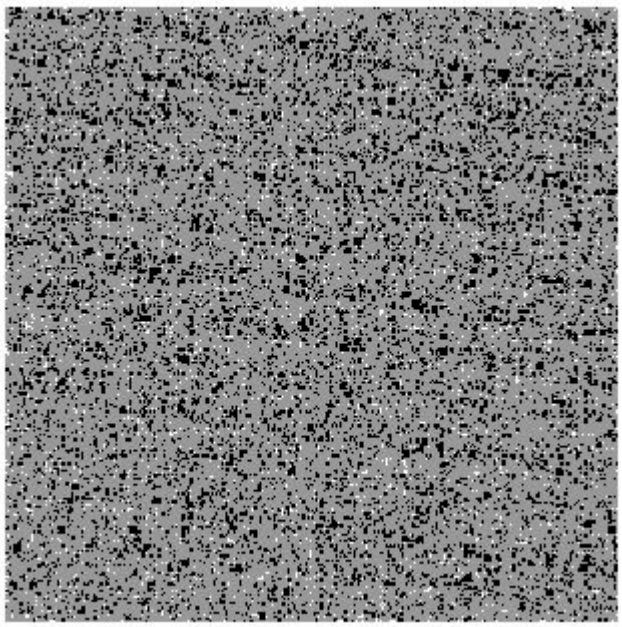


(b)

Figure 5.6 Original and Restored images of Area-4 (water), Band-6



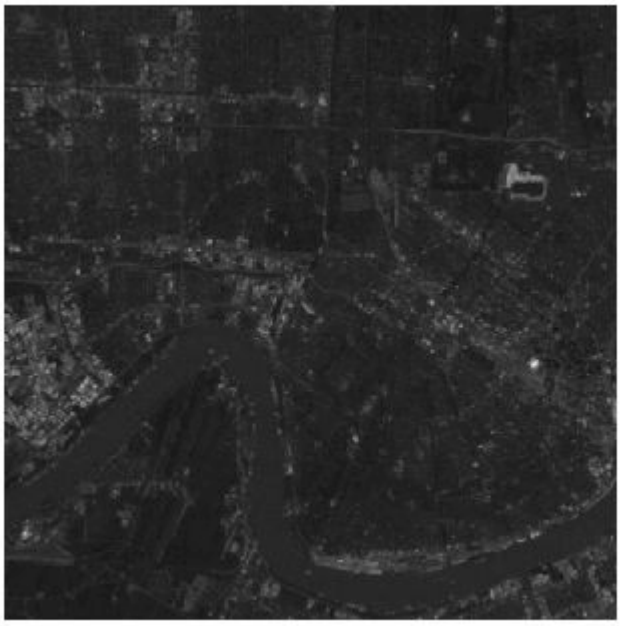
(a)



(b)

Figure 5.7 Original and Restored images of Area-4 (water), Band-7

(a)



(b)

Figure 5.8 Original and Restored images of Area-5 (urban), Band-1

(a)

(b)

Figure 5.9 Original and Restored images of Area-5 (urban), Band-2

(a)

(b)

Figure 5.10 Original and Restored images of Area-5 (urban), Band-3

(a)

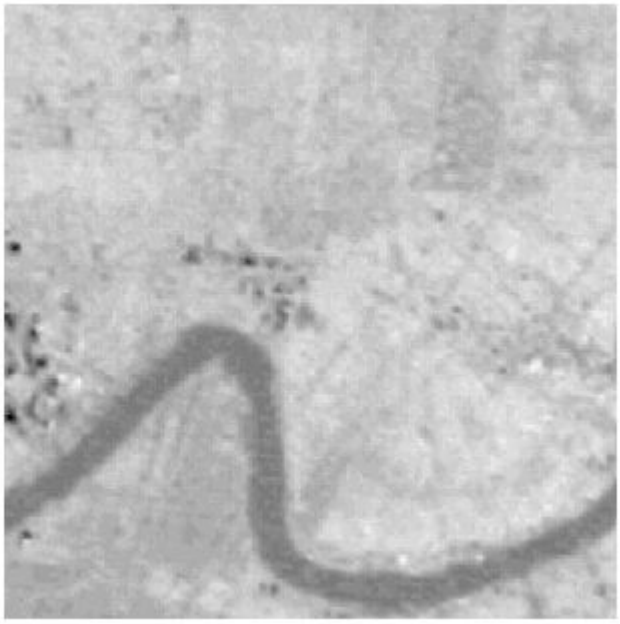
(b)

Figure 5.11 Original and Restored images of Area-5 (urban), Band-4

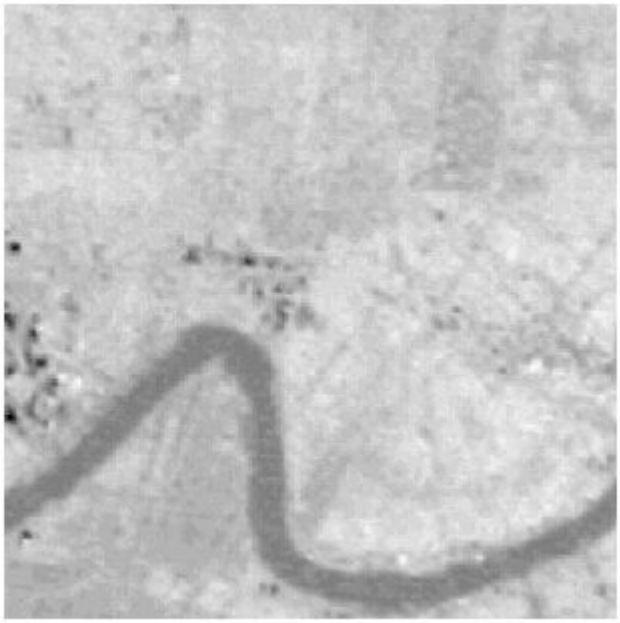
(a)

(b)

Figure 5.12 Original and Restored images of Area-5 (urban), Band-5



(a)



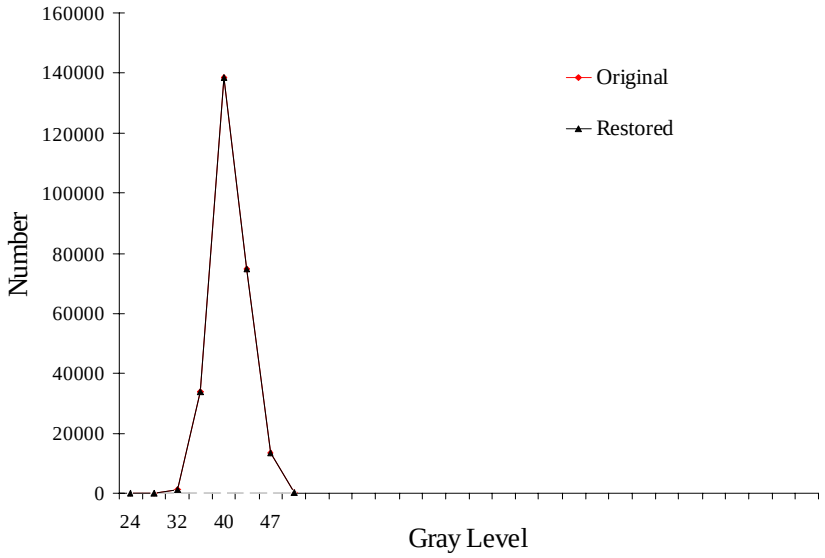
(b)

Figure 5.13 Original and Restored images of Area-5 (urban), Band-6

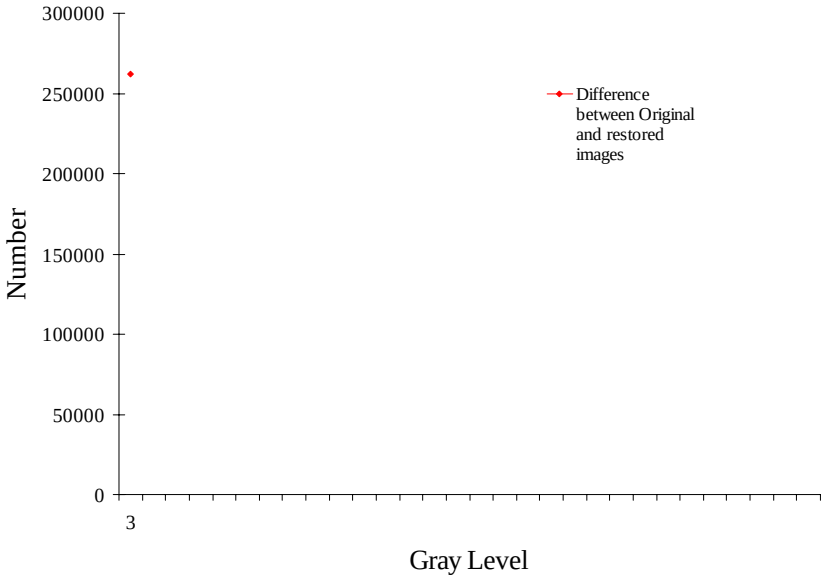
(a)

(b)

Figure 5.14 Original and Restored images of Area-5 (urban), Band-7



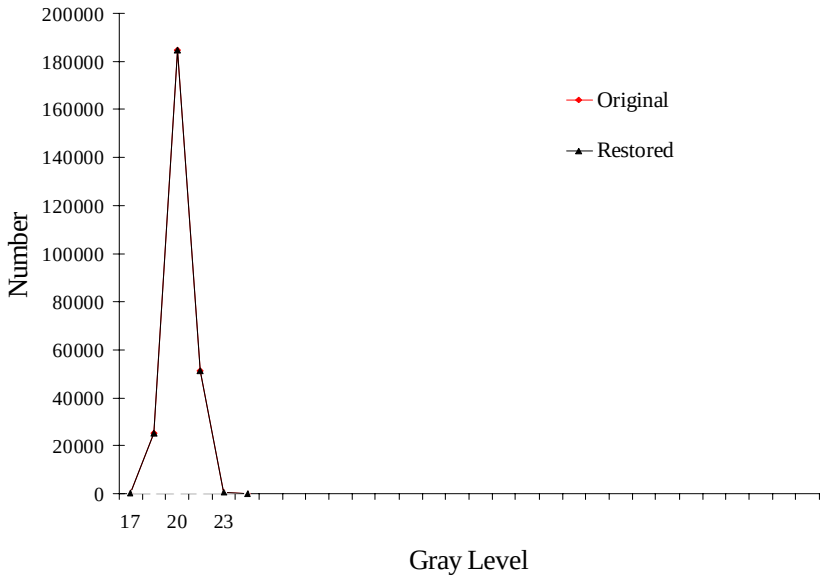
(a)



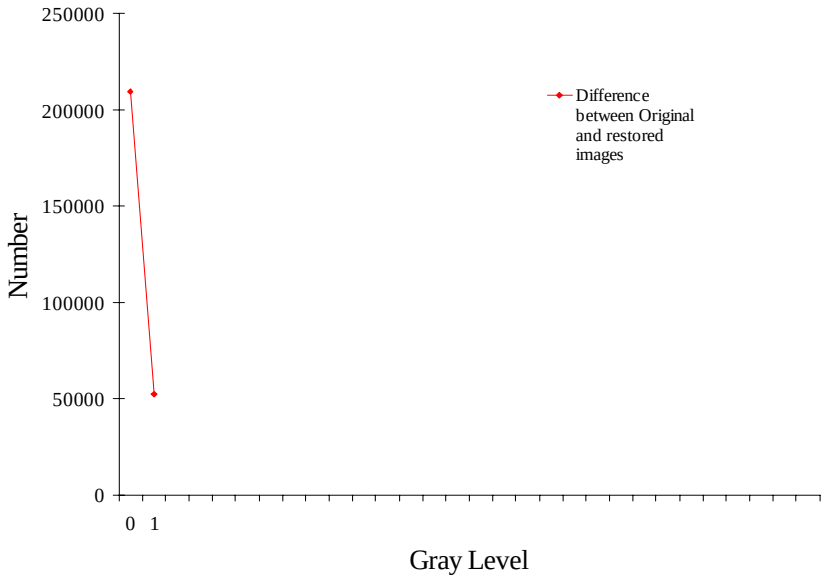
(b)

Figure 5.15 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 1

(Note: When the Original and the Restored Images have similar distribution, the two curves overlap.)

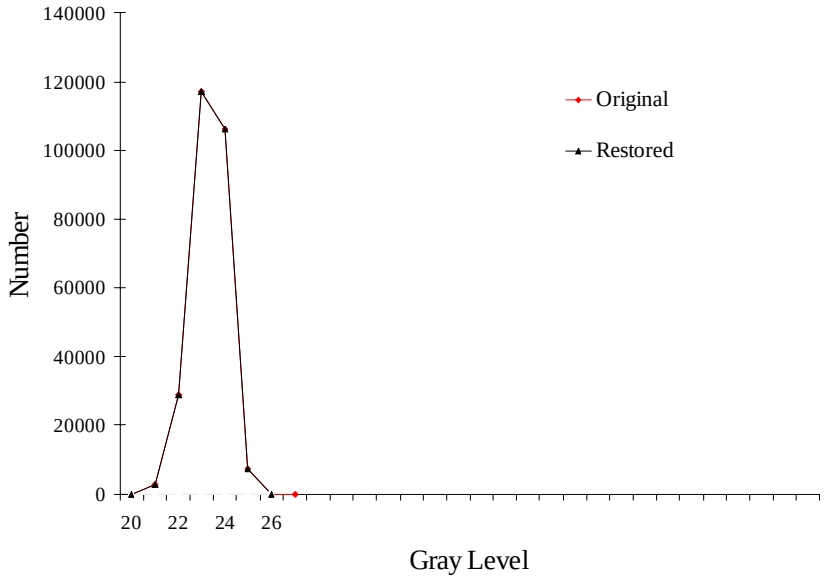


(a)

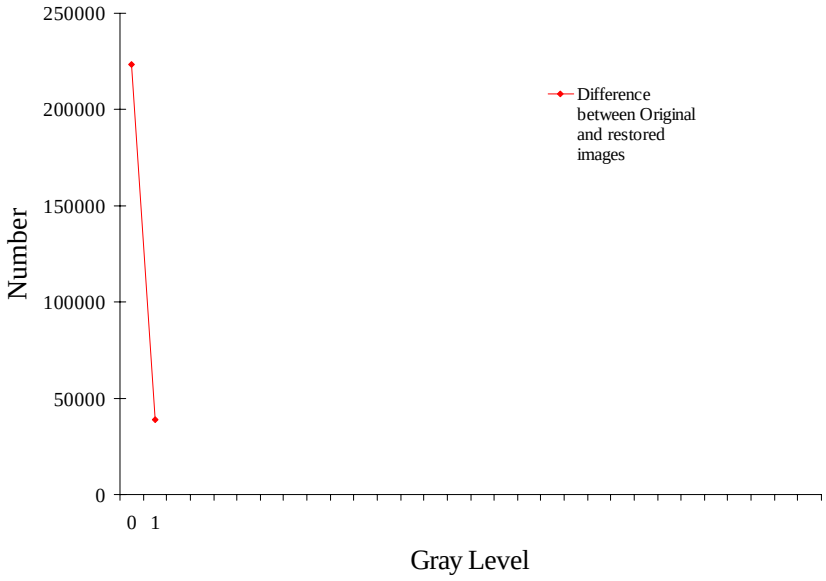


(b)

Figure 5.16 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 2

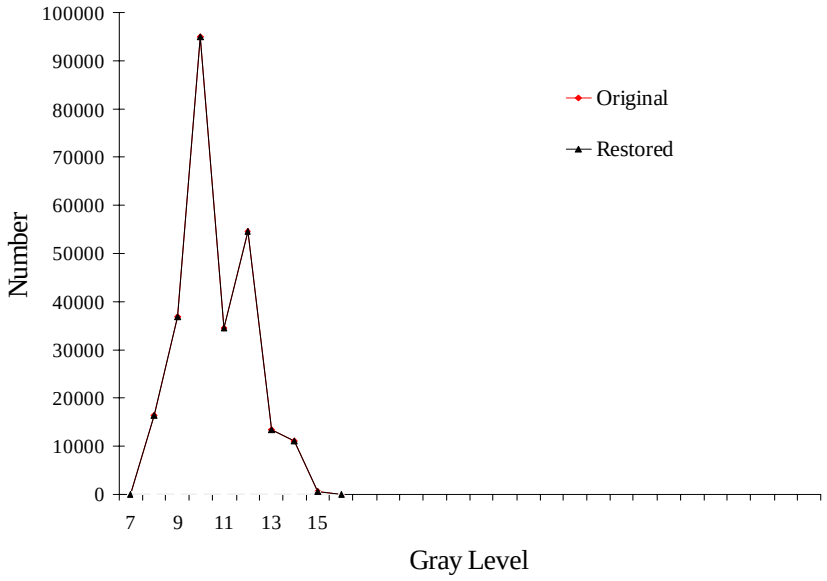


(a)

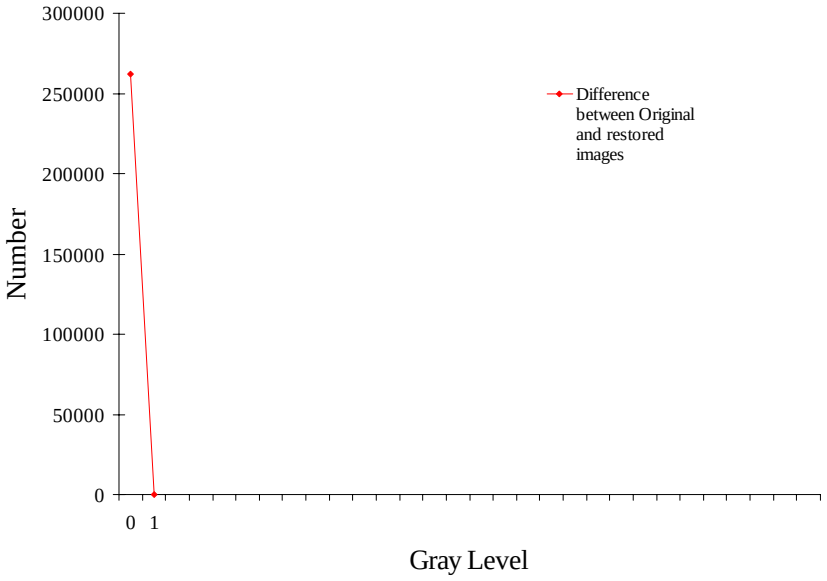


(b)

Figure 5.17 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 3

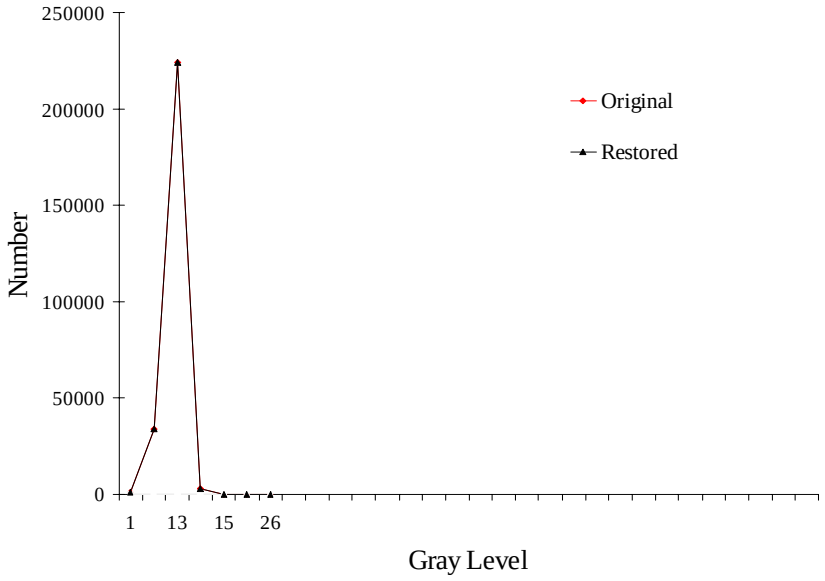


(a)

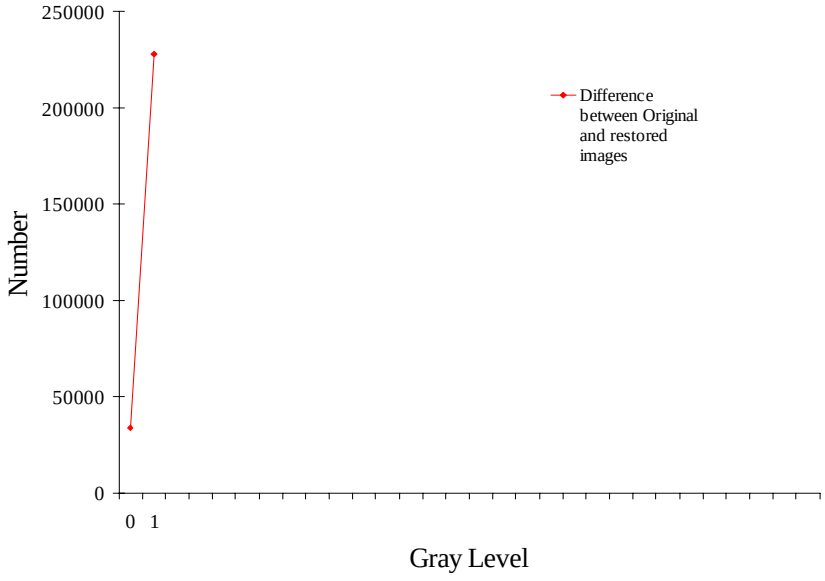


(b)

Figure 5.18 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 4

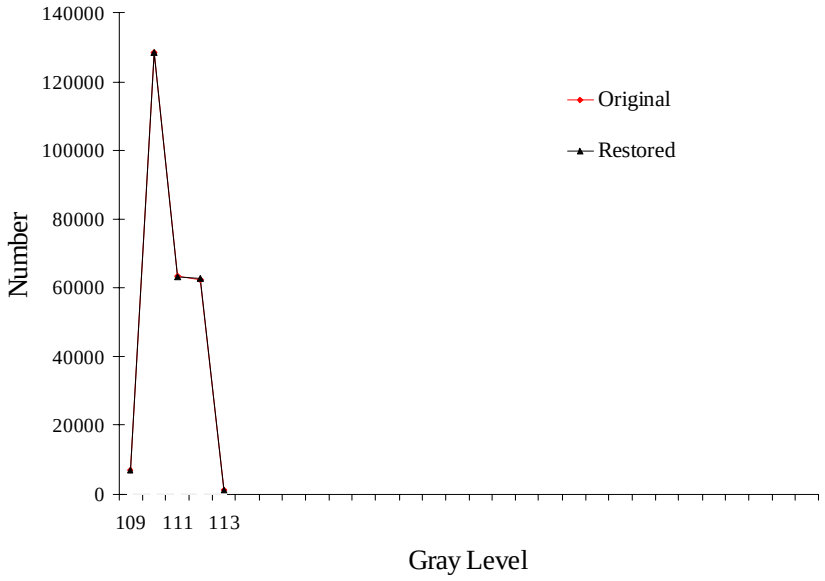


(a)

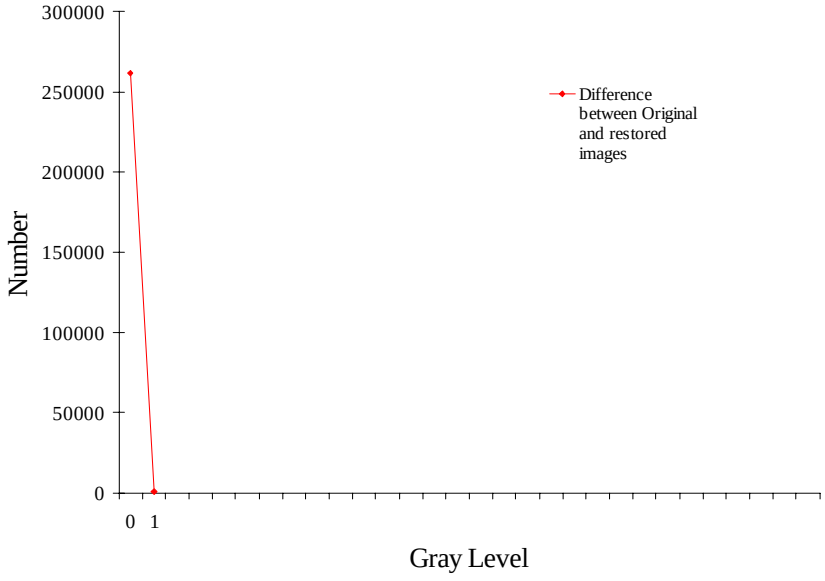


(b)

Figure 5.19 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 5

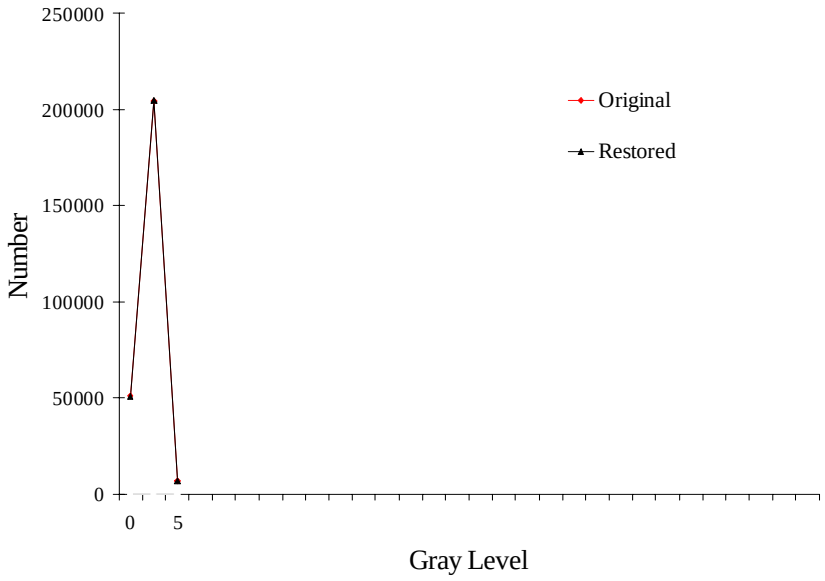


(a)

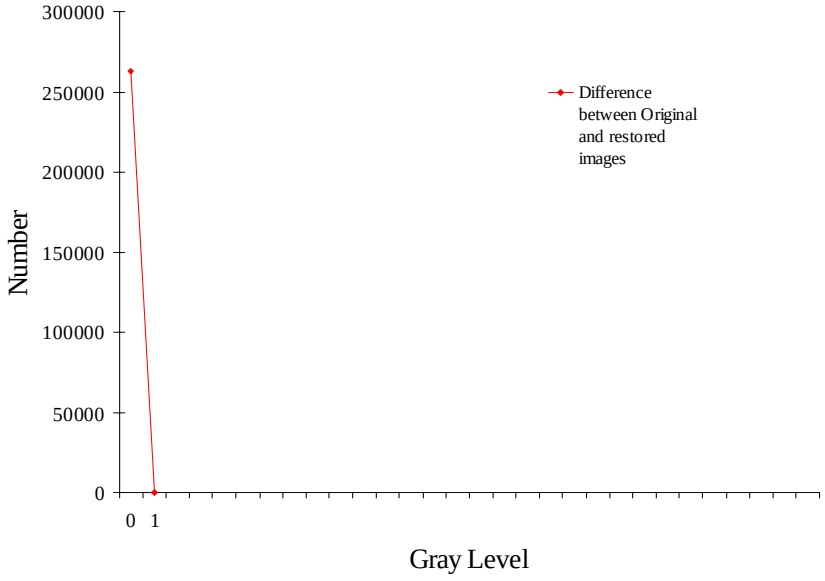


(b)

Figure 5.20 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 6

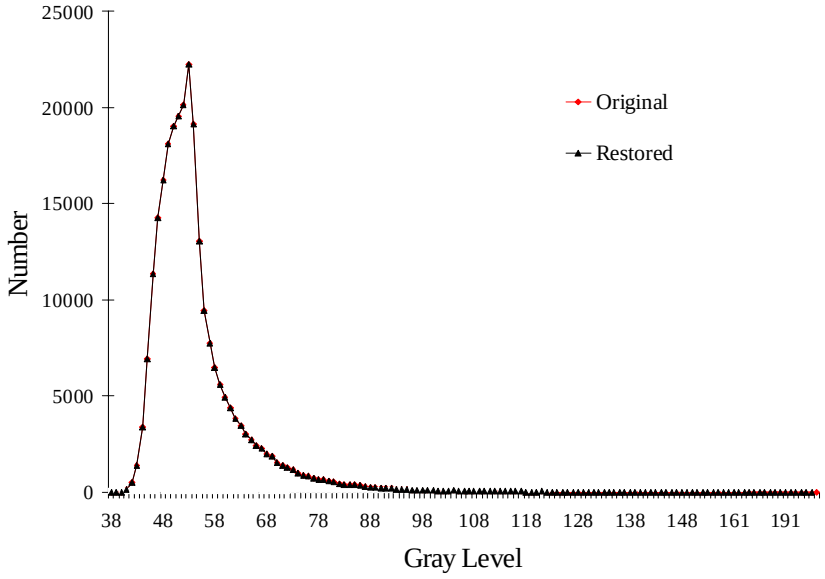


(a)

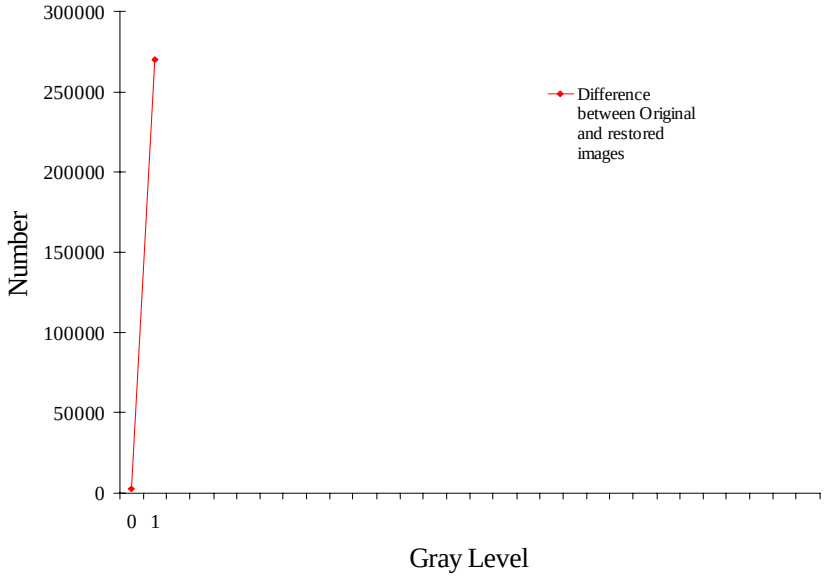


(b)

Figure 5.21 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-4 (water), Band 7

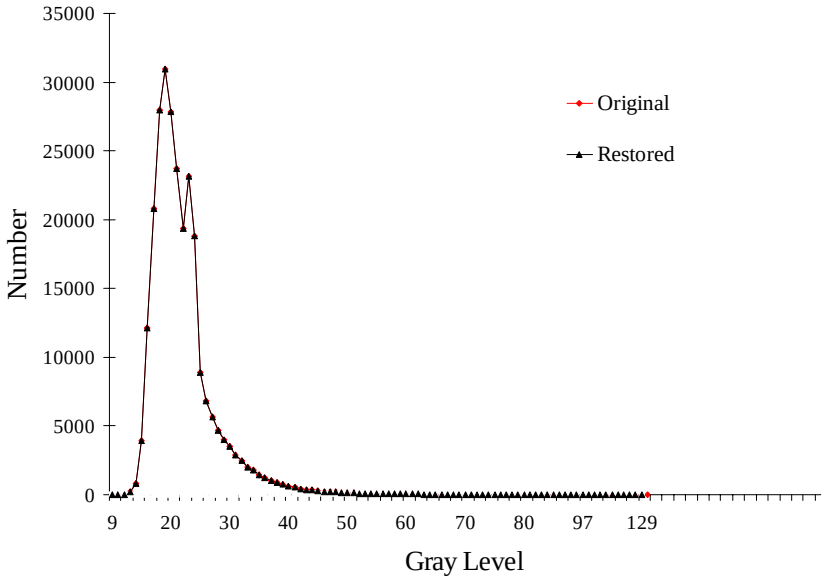


(a)

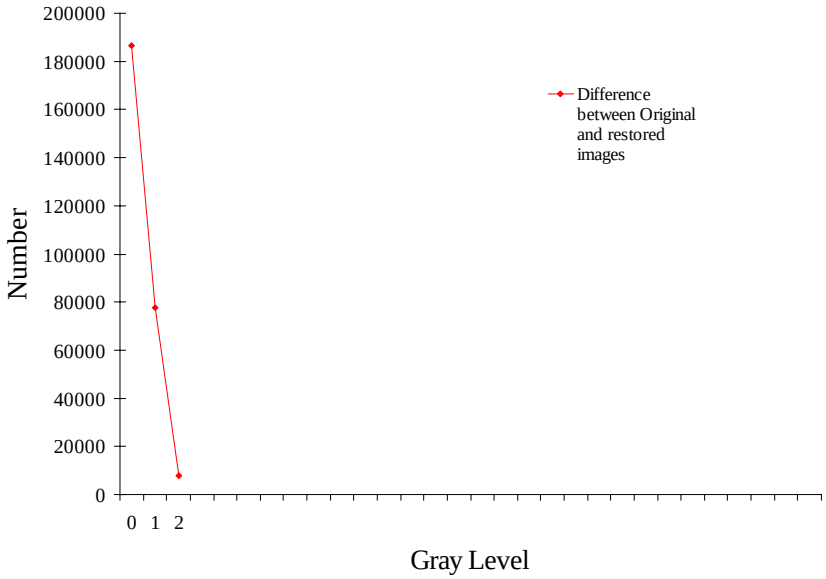


(b)

Figure 5.22 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 1

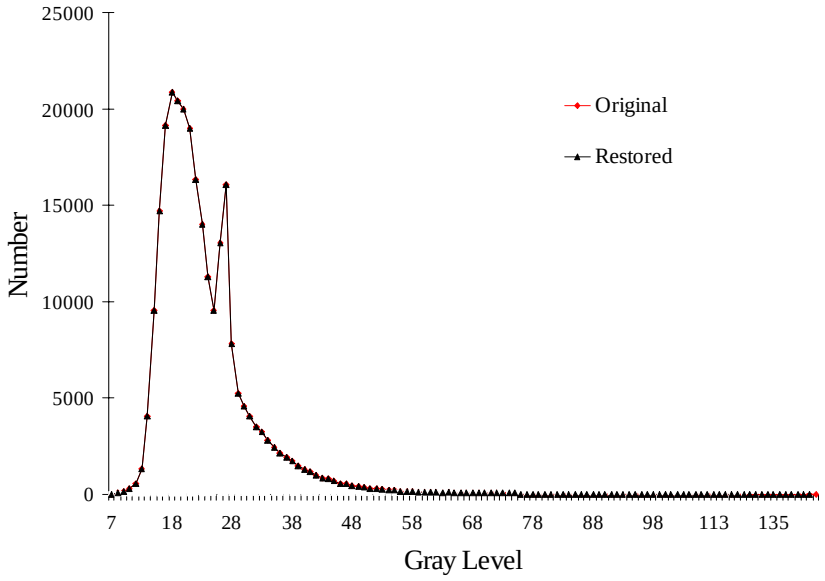


(a)

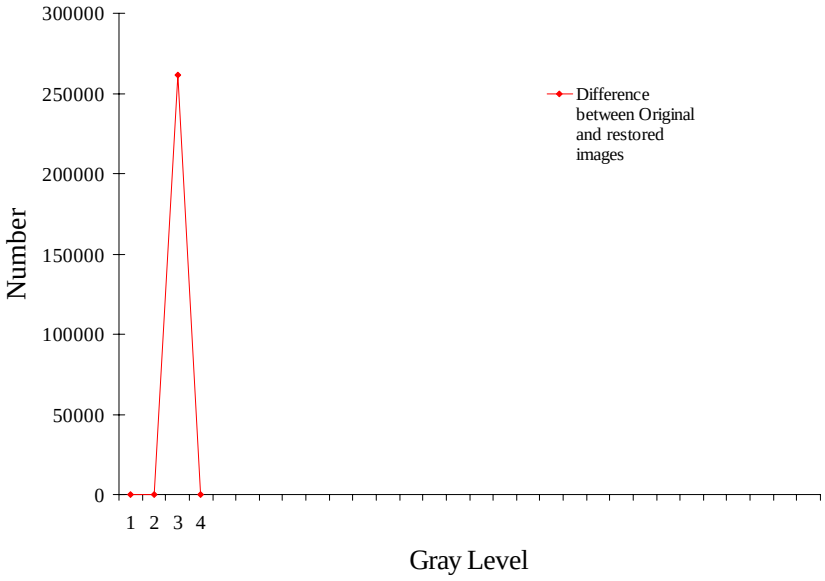


(b)

Figure 5.23 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 2

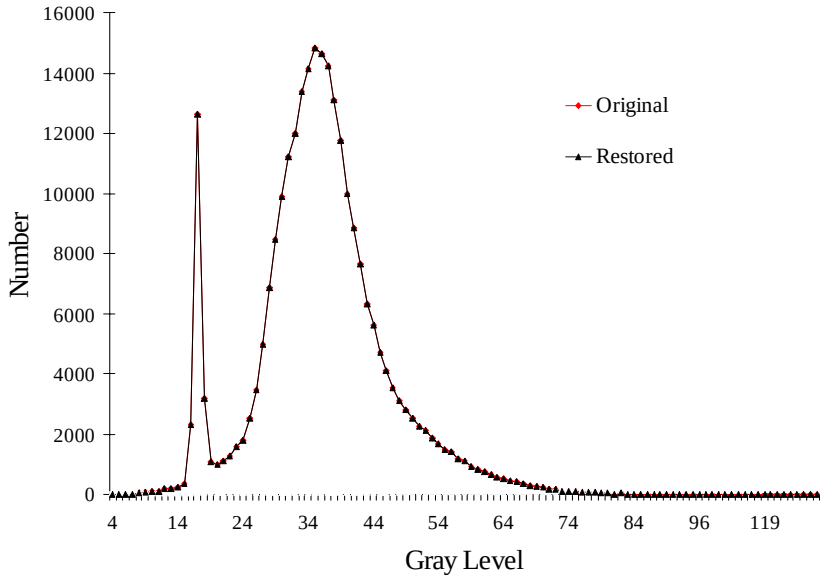


(a)

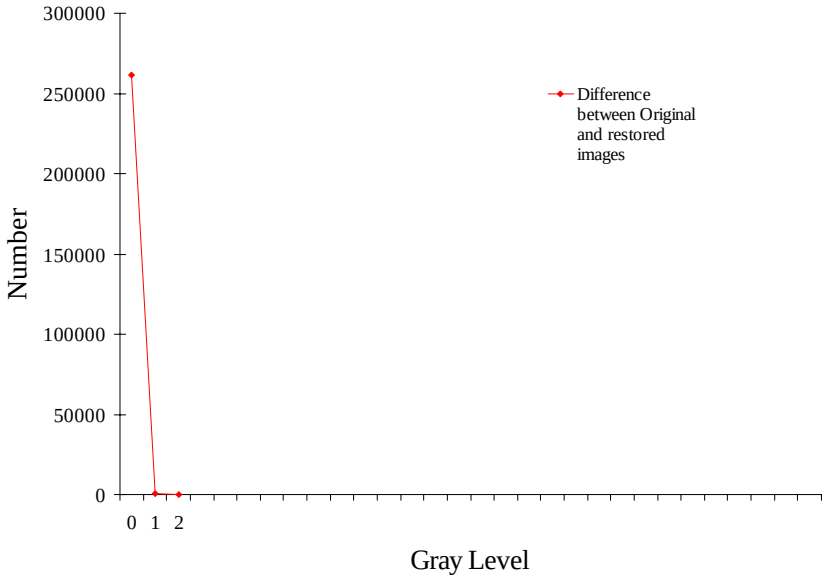


(b)

Figure 5.24 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 3

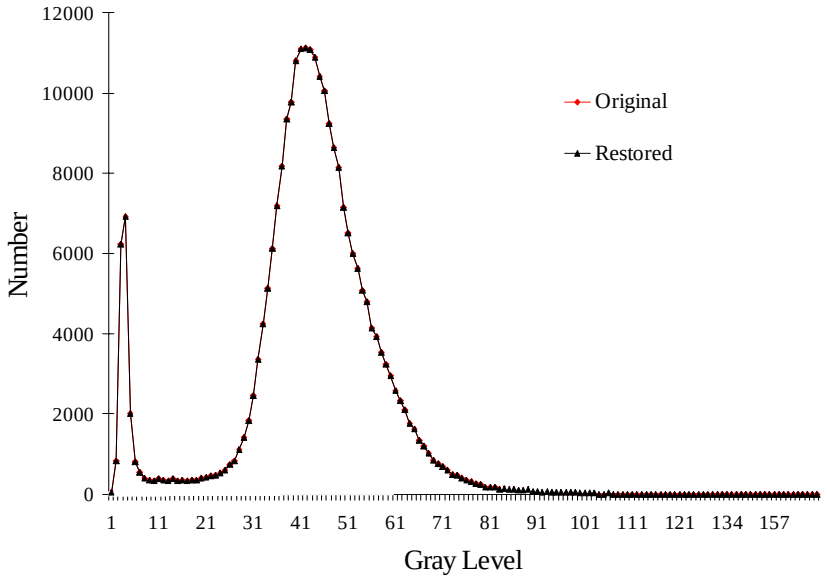


(a)

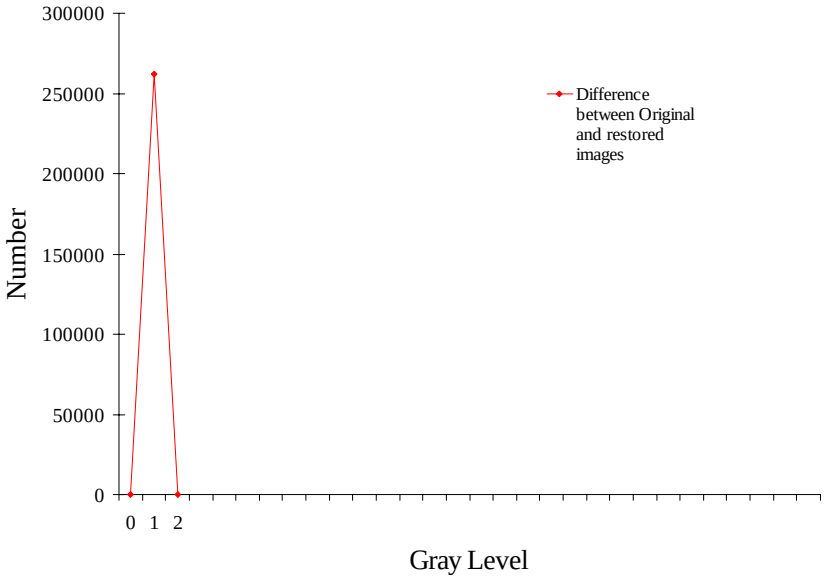


(b)

Figure 5.25 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 4

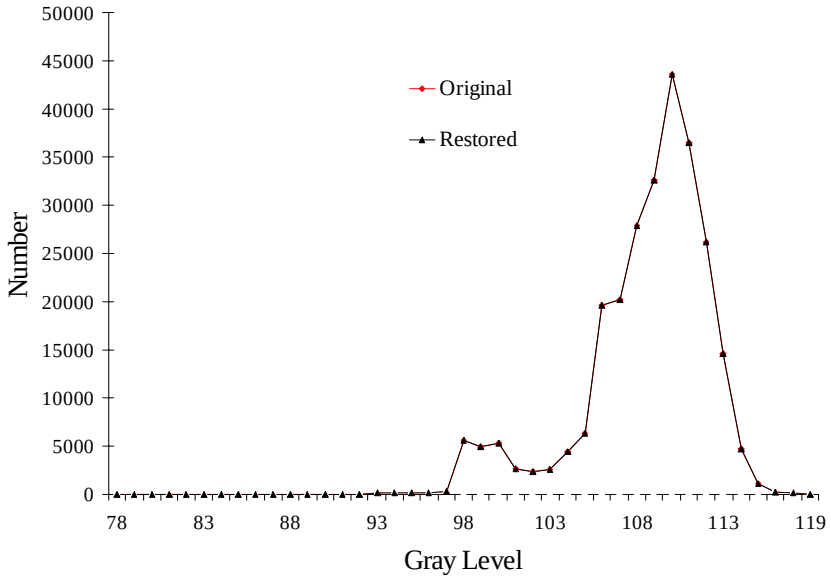


(a)

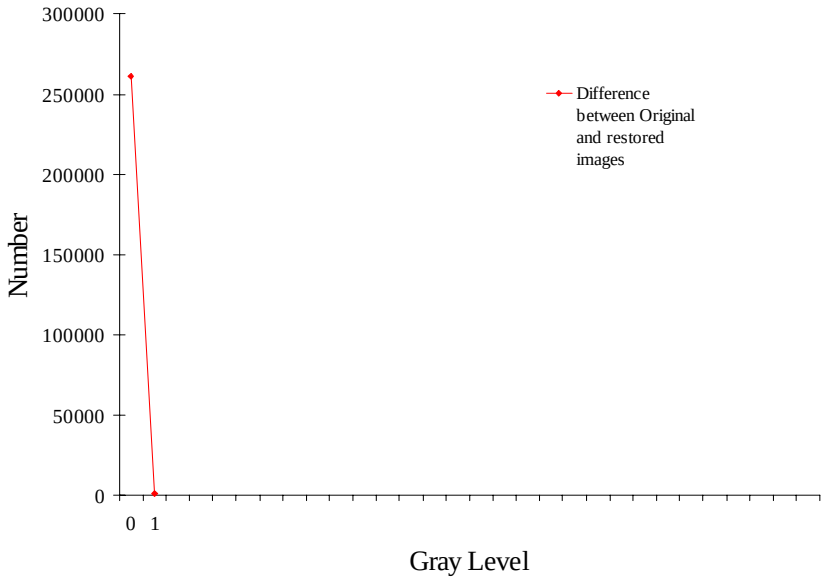


(b)

Figure 5.26 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 5

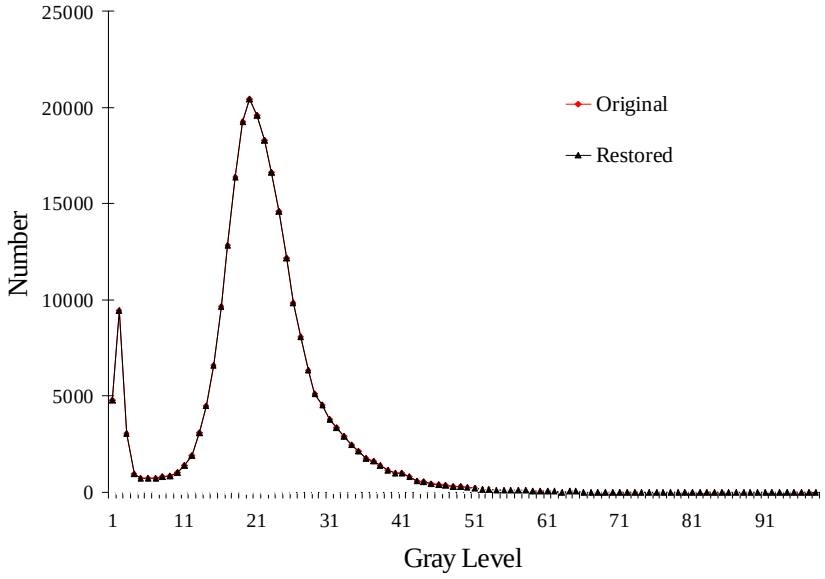


(a)

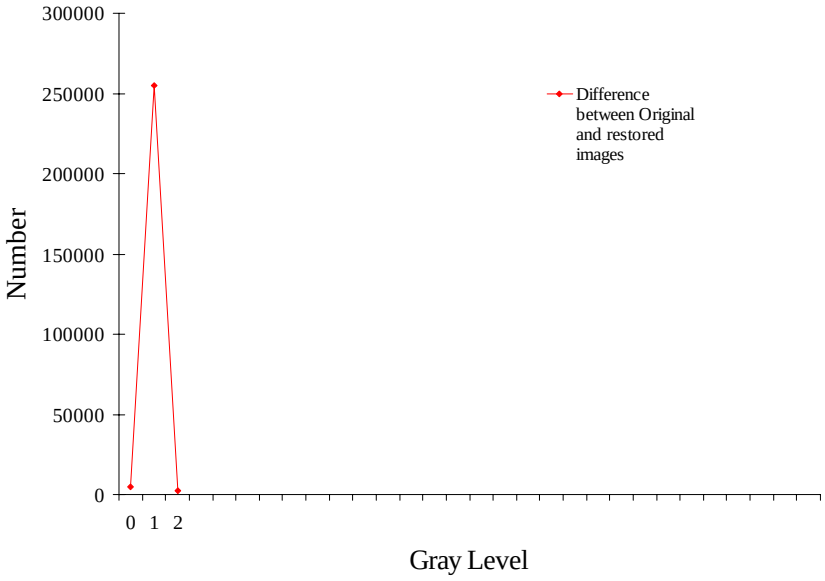


(b)

Figure 5.27 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 6



(a)



(b)

Figure 5.28 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-5 (urban), Band 7

Table 5.3 Comparison of Classification of Study Areas 1 to 4

AREA	CLASS	ORIGINAL IMAGE	RESTORED IMAGE	MATCHING
1	1	46761	46761	100%
	2	44791	44791	100%
	3	32735	32735	100%
	4	30492	30492	100%
	5	29911	29419	98.35%
	6	23453	23511	99.76%
	7	21827	21484	98.40%
	8	15357	14733	95.90%
	9	9539	9847	97%
	10	7278	7371	98.8%
2	1	61135	60927	99.66%
	2	33484	33153	99.01%
	3	28716	28776	99.79%
	4	18511	18295	98.83%
	5	27099	27316	99.20%
	6	31252	31263	99.97%
	7	18502	18499	99.98%
	8	14066	14116	99.65%
	9	12247	12236	99.91%
	10	17132	17563	97.5%
3	1	46918	46919	99.99%
	2	38798	38796	99.99%
	3	44908	44930	99.99%
	4	31240	31240	100%
	5	27958	27977	99.93%
	6	19975	19974	99.99%
	7	18414	18414	100%
	8	13006	12977	99.78%
	9	12272	12272	100%
	10	8655	8645	99.88%
4	1	100140	100215	99.93%
	2	46902	47078	99.63%
	3	25992	25976	99.94%
	4	7662	7769	98.7%
	5	847	845	99.76%
	6	463	463	100%
	7	80138	79798	99.76%

Table 5.4 Comparison of Classification of Study Areas 5 to 8

AREA	CLASS	ORIGINAL IMAGE	RESTORED IMAGE	MATCHING
5	1	4866	5011	97.1%
	2	17200	16830	97.85%
	3	25960	26546	97.74%
	4	25640	25386	99.00%
	5	39553	39872	99.20%
	6	12503	12115	96.90%
	7	16110	16028	99.49%
	8	72999	72908	99.88%
	9	12625	12564	99.52%
	10	34688	34884	99.44%
6	1	79243	79269	99.97%
	2	71161	70042	98.43%
	3	34966	36215	96.5%
	4	26548	27073	98.03%
	5	18257	17950	98.32%
	6	8349	8151	97.63%
	7	17181	17013	99.02%
	8	2389	2387	99.91%
	9	2030	2027	99.85%
	10	2020	2017	99.85%
7	1	63916	63815	99.84%
	2	48613	48613	100%
	3	37153	37153	100%
	4	29062	28880	99.37%
	5	30119	30119	100%
	6	23781	23781	100%
	7	15478	15632	99.01%
	8	6282	6246	99.43%
	9	3922	3983	98.45%
	10	3818	3922	97.28%
8	1	73426	73435	99.99%
	2	65607	65707	99.85%
	3	35869	35871	99.99%
	4	30778	30519	99.16%
	5	16806	16800	99.96%
	6	7747	7726	99.73%
	7	13745	13871	99.18%
	8	7988	8022	99.67%
	9	4729	4737	99.83%
	10	5549	5556	99.87%

Table 5.5 Comparison of Compression rate Using Fractal, Zip, and JPEG on Area 1 - 4

IMAGE	BAND	FRACTAL RATE	WINZIP RATE	JPEG RATE
1	1	32.9327	2.7363	1.9163
	2	105.7459	3.7700	5.4069
	3	21.0693	2.8132	3.9882
	4	8.3445	2.2180	1.5593
	5	5.0910	1.6608	1.1541
	6	306.2430	5.0951	3.3405
	7	8.9911	2.2150	2.9951
2	1	19.3650	2.6236	1.8453
	2	62.2374	3.3373	4.8600
	3	16.0874	2.6339	3.6164
	4	8.1391	2.0409	1.4730
	5	5.1222	1.6335	1.1391
	6	373.9572	5.0761	3.2816
	7	8.0417	2.1017	2.5694
3	1	33.0031	2.6269	1.8070
	2	100.2846	3.1546	4.5090
	3	15.3606	2.4685	3.4168
	4	5.4228	1.7425	1.4479
	5	4.8432	1.4715	0.9999
	6	364.5953	4.2200	3.0595
	7	6.8913	2.0325	2.6259
4	1	459.0963	4.4728	3.0465
	2	442.0641	6.7242	10.3186
	3	408.3240	5.9400	12.5194
	4	384.9398	5.7820	9.9728
	5	463.9717	9.9323	11.4373
	6	408.9199	7.9135	4.3885
	7	445.0662	7.1407	17.1605

Table 5.6 Comparison of Compression rate Using Fractal, Zip, and JPEG of Area 5 – 8

IMAGE	BAND	FRACTAL RATE	WINZIP RATE	JPEG RATE
5	1	5.4184	1.8058	1.1028
	2	7.5951	2.1511	2.2624
	3	5.6943	1.9014	1.7523
	4	4.9524	1.6751	1.1274
	5	4.8743	1.5207	0.9100
	6	119.4272	3.7547	2.8321
	7	5.1265	1.8034	1.8857
6	1	68.8946	3.2346	2.3047
	2	123.5363	4.1058	6.5590
	3	66.0146	3.7568	5.6329
	4	10.1261	2.1938	2.1423
	5	9.2379	2.0472	1.6469
	6	386.0736	5.5835	3.4845
	7	25.6928	2.6995	4.3338
7	1	30.6996	2.8148	1.9518
	2	96.0234	3.7874	5.7348
	3	23.9423	2.8716	4.3212
	4	6.8871	2.0467	1.7306
	5	5.3067	1.7175	1.3036
	6	346.7513	4.8248	3.3900
	7	10.2564	2.2950	3.1949
8	1	47.5329	2.9317	2.0588
	2	124.3932	4.4301	6.6262
	3	36.8749	3.1855	4.8268
	4	7.7017	2.1919	1.5657
	5	6.4194	1.8495	1.5576
	6	338.2503	4.7288	3.2698
	7	13.2785	2.4829	3.4655

Table 5.7 The Relationship between Fractal Compression rate and Complexity for Study Area 1-4

AREA	BAND	COMPRESSION RATE	F-DIMENSION (TRIANGULAR)	C.V.	ENCODE TIME(m's'')	DECODE TIME(m's'')
1	1	32.9327	2.5882	1.197	immediate	0'27.12''
	2	105.7459	2.5509	0.795	immediate	0'27.60''
	3	21.0693	2.508	1.540	immediate	0'27.47''
	4	8.3445	2.5832	1.828	0'29.61''	0'28.84''
	5	5.0910	2.4903	5.570	0'55.88''	0'28.80''
	6	306.2430	2.2295	0.710	immediate	0'26.40''
	7	8.9911	2.4858	2.759	0'31.50''	0'28.08''
2	1	19.3650	2.5472	1.307	immediate	0'28.00''
	2	62.2374	2.529	0.959	immediate	0'26.56''
	3	16.0874	2.4886	1.551	immediate	0'27.54''
	4	8.1391	2.4413	3.218	0'36.36''	0'28.21''
	5	5.1222	2.452	5.576	0'58.87''	0'29.97''
	6	373.9572	2.4469	0.530	immediate	0'26.10''
	7	8.0417	2.4623	2.727	0'36.38''	0'28.64''
3	1	33.0031	2.5928	1.122	immediate	0'27.28''
	2	100.2846	2.5917	0.902	immediate	0'26.86''
	3	15.3606	2.5589	1.565	0'23.4''	0'27.88''
	4	5.4228	2.5459	4.081	0'53.34''	0'29.05''
	5	4.8432	2.5237	6.917	1'0.27''	0'29.24''
	6	364.5953	2.5159	0.758	immediate	0'26.18''
	7	6.8913	2.5342	2.694	0'44.85''	0'28.12''
4	1	403.9199	2.55	0.415	immediate	0'26.22''
	2	411.5290	2.5402	0.250	immediate	0'26.04''
	3	407.0559	2.5427	0.317	immediate	0'26.64''
	4	397.1879	2.5954	0.555	immediate	0'27.60''
	5	500.2748	2.5775	0.277	immediate	0'26.64''
	6	390.0952	2.4068	0.233	immediate	0'26.64''
	7	380.4702	2.5842	0.276	immediate	0'27.09''

Table 5.8 The Relationship between Fractal Compression rate and Complexity for Study Area 5-8

AREA	BAND	COMPRESSION RATE	F-DIMENSION (TRIANGULAR)	C.V.	ENCODE TIME(m's'')	DECODE TIME(m's'')
5	1	5.4184	2.8002	3.523	0'49.77''	0'29.40''
	2	7.5951	2.7942	2.173	0'36.08''	0'28.56''
	3	5.6943	2.806	2.948	0'47.45''	0.28.86''
	4	4.9524	2.7643	3.842	0'52.84''	0'30.60''
	5	4.8743	2.6617	5.908	0'51.75''	0'29.92''
	6	47.9239	2.5143	1.447	immediate	0'27.84''
	7	5.1265	2.8209	3.123	0'49.92''	0'29.12''
6	1	68.8946	2.6405	0.825	immediate	0'27.27''
	2	123.5363	2.6079	0.663	immediate	0'27.25''
	3	66.0146	2.5881	0.905	immediate	0'28.00''
	4	10.1261	2.5437	2.483	0'25.20''	0'29.07''
	5	9.2379	2.5328	3.486	0'26.91''	0'29.40''
	6	386.0736	2.5111	0.438	immediate	0'27.54''
	7	25.6928	2.5907	1.600	immediate	0'28.05''
7	1	30.6996	2.6045	1.052	immediate	0'26.62''
	2	96.0234	2.5884	0.734	immediate	0'26.22''
	3	23.9423	2.5445	1.327	immediate	0'27.25''
	4	6.8871	2.5477	2.372	0'37.80''	0'29.26''
	5	5.3067	2.5909	4.436	0'54.12''	0'29.21''
	6	346.7513	2.4271	0.711	immediate	0'26.68''
	7	10.2564	2.6128	2.061	0'26.28''	0'28.08''
8	1	47.5329	2.629	0.999	immediate	0'26.97''
	2	124.3932	2.5657	0.675	immediate	0'26.52''
	3	36.8749	2.539	1.239	immediate	0'27.59''
	4	7.7017	2.6077	1.783	0'32.00''	0'28.71''
	5	6.4194	2.5469	4.518	0'45.20''	0'28.28''
	6	338.2503	2.5048	0.561	immediate	0'26.78''
	7	13.2785	2.5335	2.191	0'22.00''	0'27.81''

Table 5.9 The Relationship between Fractal Compression rate and Resolution

AREA	BAND	F-DIMENSION (ORIGINAL)	F-DIMENSION (RESAMPLE)	COMPRESSION RATE(ORIGINAL)	COMPRESSION RATE(RESAMPLE)
4	1	2.8002	2.6895	403.9199	459.0963
	2	2.7942	2.6760	411.5290	442.0641
	3	2.8060	2.6776	407.0559	408.3240
	4	2.7643	2.5917	384.9398	397.1879
	5	2.6617	2.5207	463.9717	500.2748
	6	2.5143	2.4183	390.0952	408.9199
	7	2.8209	2.7142	380.4702	445.0662
5	1	2.5500	2.4058	5.4184	20.6901
	2	2.5402	2.3984	7.5951	21.8508
	3	2.5427	2.4013	5.6943	21.8217
	4	2.5954	2.4624	4.9524	21.9257
	5	2.5775	2.4464	4.8743	25.8703
	6	2.4068	2.3459	47.9239	119.4272
	7	2.5842	2.4688	5.1265	22.1874

Table 5.10 Average Compression Rates of Images of Study Areas

STUDY AREA	COMPRESSION RATE
1	30.362401
2	19.832137
3	27.634278
4	433.910347
5	5.610168
6	50.583721
7	28.852589
8	39.366779

Table 5.11 Average Compression Rates of Separate Bands

OVERALL BAND	COMPRESSION RATE
1	33.978055
2	88.545123
3	26.434775
4	7.367679
5	5.842096
6	319.328277
7	11.182619

CHAPTER 6

CONCLUSION

Remote sensing images contain huge amount of geographical information and reflect the complexity of geographical features, landscape, land cover types, and spatial structures. As the means of observing and describing geographical phenomena, the rapid development of remote sensing has provided progressively more geographical information. On one hand, the enormous information is very useful in a variety of applications; on the other hand, the sheer bulk of this information has increased beyond what can be analyzed and used efficiently and effectively. This uneven increase in the technologies of gathering and analyzing information has created difficulties in its storage, transfer, and processing. Fractal geometry provides a means of describing and analyzing the complexity of different geographical features in remotely sensed images. It also provides a more powerful tool to compress the remote sensing data than the traditional methods. This study suggests, for the first time, the implementation of this usage of fractals to remotely sensed images.

In this study, the requirement for more powerful compression tools to expediate storage and analysis for remotely sensed images was analyzed. Fractal compression and decompression methods were developed, described, and discussed from concepts and algorithms, to the final application of remotely sensed imagery. The compression results were compared with the original data in different ways, both visually and quantitatively. Fractal compression results were also analyzed with other approaches including image processing techniques, image classification method, and statistical tools. Findings of this study are as follows:

- Fractal techniques can be applied to compress and decompress remote sensing images. Fidelity and efficiency of the restored images by fractal techniques are better than those of the previous image compression methods.
- In comparison with the most commonly used compression methods, JPEG and WinZip, fractal method has a higher compression rate.
- Fractal compression rates can be affected by the spatial complexity of remote sensing image. The more complex the images, the lower the compression rates and the longer the compressing time. So fractal compression rate can be used as an indicator of complexity of geographical features reflected by remotely sensed images. This relationship between complexity and compression rate was verified with both statistical tools and fractal dimensions.
- Fractal compression rates can also be affected by spatial resolution of remote sensing images. In particular, the lower the pixel resolution, the higher the compression rate. When resolution decreases, the spatial complexity also decreases so that the compression rate will increase. This result is consistent with the fact that complexity will affect the fractal compression rate.

It can be said that this study advances the fractal technique by applying it to remote sensing image compression and restoration. Besides the application of the methods in compression, fractals also provides a new usage as the compression rate can be a measure of the complexity of geographical features in remote sensing images.

This study is significant in that it establishes the relationship between the landscapes and the compression rate. This relationship was clearly revealed and a better understanding was obtained about how the geographical phenomena affect the image

compression rates and processing times. Smooth land cover types have higher compression rates and less processing time than those rough ones while certain rougher ones have lower compression rates. In this study, water body (Lake Pontchartrain) has very high compression rates in all bands. Also, the thermal infrared band in all the study areas has high compression rate because its spatial resolution is lower than other bands so that it appears smoother than others. The urban area in New Orleans has the most complex geographical features, consequently the compression rates of the corresponding images are the lowest.

In addition to the fractal's use in measuring, describing, and simulating the roughness of landscapes in geography, fractal techniques were usefully developed in the remotely sensed image compressions of this study. Moreover, the compression technique can be seen as a new method of measuring the diverse landscapes and geographical features. This study has clearly introduced a new and advantageous passageway for fractal applications and their vastly important applications in geography and remote sensing.

As a technique of image processing, fractal image compression applied to remotely sensed images has great potential and immediate impact upon extending the horizon of image compression technology and methods. This study seeks to enable the scientist to more efficiently and effectively understand the geographical moment of his/her interest as well as maintain a record of this instant in geographic time and space. Higher compression rates and image fidelity were obtained in this study, saving resources of time, man hours, and hardware. The efficiency of the algorithms was also tested. So it has become a complete set of the new technique.

This study has successfully demonstrated that fractal image compression and analysis is a powerful and resourceful tool in remote sensing image processing and geographical feature complexity measurement. This technique is in its infancy and without doubt has great potential to develop in both the technique itself and valuable applications in geography. Some considerations about possible future studies as related to the current one are listed below.

1. Fractal compression rate can be an indirect measure of the complexity of geographical features in remote sensing images. This has been established in this study. To grow as an index of complexity, more images having different degree of complexity will need to be analyzed, with the intention that the compression rates of 'pure' geographical features or land cover types can be obtained.
2. This relationship can be profoundly revealed in the aspect of spatial resolution or scale of remote sensing images. The research about the function of compression rates as an index of complexity can be intensified and extended. As it has been seen, the compression rates are related consistently to the spatial resolutions of images. Therefore, more quantitative research can further be focused and conducted. Given an image of a specific land cover type, compression rates for various spatial resolutions can be tested so that the relationship between spatial resolution and compression rates will be more clearly revealed and understood. In this aspect, multi-resolution and multi-scale analysis can be a valuable reference.

3. The research can be extended to more kinds of remotely sensed images from different sources and platforms. For example, SPOT, AVHRR, RADAR, and IKONOS images, which have different spatial, spectral, and temporal resolution, can be tested to further develop fractal tools as well as to intensify and expand the knowledge with reference to the properties and complexities of different landscapes from remotely sensed images.
4. Last but not least, the compression and decompression algorithms still need refining to meet the various necessities of multiple applications. Some applications may need faster encoding and decoding, such as data transferring through internet. Others may require high compression rates. A number of other applications may emphasize fidelity, e.g. data calibration. Accordingly the algorithms need to be more flexible and more robust to deal with different situations.

REFERENCES

Abonsleman, G. P., Marchllin, M.W., and Hunt, B.R., 1995. Compression of Hyperspectral Imagery Using the 3-D DCT and Hybrid DPCM/DCT. *IEEE Transactions on Geoscience and Remote Sensing* 33(1): 26-34

Anh. V., Maeda, J., Ishizaka, T., Suzuki, Y., and Tieng, Q., 1997. Two-dimensional fractal segmentation of natural images. *Lecture Notes in Computer Science: Image Analysis and Processing ICIAP '97, 9th International Conference on Image Analysis and Processing* 286-293.

Argialas, D., 2000. Extraction and Delineation of Alluvial Fans from Digital Elevation Models and Landsat Thematic Mapper Images. *Photogrammetric Engineering and Remote Sensing* (66)9: 1093-1102.

Arlinghaus, S., 1993. Central Place Fractals: Theoretical Geography in an Urban Setting, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp 211-246.

Aronoff,S., 1986. *Geographic Information Systems: A Management Perspective*. WDL Publications, Ottawa, Canada.

Atkinson, P. M., and Tate, N.J.. eds., 1999. *Advances in Remote Sensing and GIS Analysis*. John Wiley & Sons Ltd, New York.

Bailey, K., Frohn, R., Beck, R., and Price, M., 2001. Remote Sensing Analysis of Wild Rice Production Using Landsat 7 for the Leech Lake Band of Chippewa in Minnesota. *Photogrammetric Engineering and Remote Sensing* 67(2): 189-192.

Barnsley, M. F., 1988. *Fractals Everywhere*. New York: Academic Press Inc., London.

Barnsley, M. F., Sloan, A., 1988. A Better Way to Compress Images. *Byte* 88(1):25-32.

Barnsley, M. F., and Hurd, L. P. 1993. *Fractal Image Compression*. AK Peters, Ltd., Wellesley, Massachusetts.

Bates, R. H. T., and McDonnell, M. J., 1986. *Image Restoration and Reconstruction*. Clarendon Press, New York.

Batty, M. 1985. Fractals - Geometry between Dimensions. *New Scientists*, April 4: pp31-35.

Batty, M., and Longley, P., 1994. *Fractal Cities*. Academic Press, San Diego.

Belloulata, K, and Konrad, J., 2002. Fractal image compression with region-based functionality. *IEEE Transactions on Image Processing*, 11(4): 351-362

Beltran, C. M., and Belmonte, A. C. 2001. Irrigated crop area estimation using Landsat TM imagery in La Mancha, Spain. *Photogrammetric Engineering and Remote Sensing* 67(10): 1177-1184.

Bindschadler, R. and vornberger, P., 1992. Interpretation of Sar Imagery of the Greenland Ice Sheet Using Co-Registered TM Imagery. *Remote Sensing of Environment* 42(3): 167-176.

Bindschadler, R., and Vornberger, P., 2000. Detecting ice-sheet topography with AVHRR, RESURS-01, and Landsat TM imagery. *Photogrammetric Engineering and Remote Sensing* 66(4): 417-422.

Braga, C. Z. F., Setzer, A. W., and Lacerda, L., D., 1993. Water Quality Assessment with Simulaneous Landsat TM Data at Guanabara Bay, Rio DeJaneiro, Brazil. *Remote Sensing of Environment* 45(1): 95-106.

Burrough, P., 1993. Fractals and Geostatistical Methods in Landscape Studies, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. Pp87-121.

Caranza, E. J. M., and Hale,M., 2002. Mineral imaging with Landsat Thematic Mapper data for hydrothermal alteration mapping in heavily vegetated terrane. *International Journal of Remote Sensing* 23(22): 4827-4852.

Casacchia, R., Salvatori, R., and Cagnati, A., 2002. Field reflectance of snow/ice covers at Terra Nova Bay, Antarctica. *International Journal of Remote Sensing* 23(21): 4653-4668.

Chen, T. M., 1987. Information Content Analysis of Landsat Image Data for Compression. *IEEE Transactions on Geoscience and Remote Sensing*, Vol. GE-25(4): 499-501.

Chica-Olmo, M., Abarca, F., and Rigol, J. P., 2002. Development of a Decision Support System based on remote sensing and GIS techniques for gold-rich area identification in SE Spain. *International Journal of Remote Sensing* 23(22): 4801-4814.

Clarke, K. C., 1986. Computation of the Fractal Dimension of Topographic SurfacesUsing the Triangular Prism Surface Area Method. *Computers and Geosciences*, 12(5): 713-722.

Clarke, K. C., 1993. One Thousand Mount Everests, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp265-281.

Cola, L., 1993. Multifractals in Image Processing and Process Imaging, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp282-304.

Collins, J. B., and Woodcock, C.E., 1996. An Assessment of Several Linear Change Detection Techniques for Mapping Forest Mortality Using Multi-temporal Landsat TM Data. *Remote Sensing of Environment* 56(1): 66-77.

Colwell, R. N., ed. –in chief, 1983. *Manual of Remote Sensing, 2nd ed*. American Society of Photogrammetry, Falls Church, VA.

Davoine, F., Antonini, M., Chassery, J. M., and Barlaud, M., 1996. Fractal Image Compression Based on Delaunay Triangulation and Vector Quantization. *IEEE Transactions on Image Processing*, 5(2): 338-346.

Destival I., 1986. Mathematical Morphology Applied to Remote Sensing. *Acta Astronautics* 13(6/7): 371-385.

Dhakal, A., Amada, T., Aniya, M., and Sharma, R., 2002. Detection of Areas Associated with Flood and Erosion Caused by a Heavy Rainfall Using Multitemporal Landsat TM data. *Photogrammetric Engineering and Remote Sensing* 68(3): 233-240.

Drury, S.A., 1990. *A Guide to Remote Sensing*. Oxford Science Publication, New York.

Dutton, G. H., 1981. Fractal enhancement of cartographic line detail. *The American Cartographer* 8(1):23-40.

Ehlers, M., 1990. Remote Sensing and GISs towards Integrated Spatial Information Processing. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 763-765.

Eiumnoh, A. and Shrestha, R. P., 2000. Application of DEM data to Landsat image classification: evaluation in a tropical wet-dry landscape of Thailand. *Photogrammetric Engineering and Remote Sensing* 66(3): 297-304.

Eichmann, G., Lu, C., Zhu, J., and Roytman, L., 1989. *A New Transform Method for Image Data Compression. in SPIE* Vol. 1099 Advances in Image Compression and Automatic Target Recognition, pp 134-141.

Ekstrand, S. P., 1990. Detection of Moderate Damage on Norway Spruce Using TM and Digital Stand Data. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 685-692.

Elach, C., 1987. *Introduction to the Physics and Techniques of Remote Sensing*. John Wiley & Sons, New York.

Esri Inc, 2001. *Arc User, The magazine for ESRI Software Users*. ESRI, Redland, California.

Fisher, Y., 1992. Fractal Image Compression Using Iterated Transforms, in *Image and Text Processing*, edited by Storer, J. A., Kluwer Academic, Boston. pp 34-61.

Fisher, Y., 1997. *Fractal Image Compression: Theory and Application*. Springer-Verlag, New York.

Foresman, T. W., ed., 1998. *The History of Geographic Information Systems: Perspectives from the Pioneers*. Prentice Hall PTR, Upper Saddle River, NJ.

Franklin, J., Woodcock, C. E., and Warbington, R., 2000. Multi-attribute vegetation maps of forest service lands in California supporting resource management decisions. *Photogrammetric Engineering and Remote Sensing* 66(10): 1209-1218.

Franklin, S. E., Peddle, D. R., and Dechka, J., A., 2002. Evidential reasoning with Landsat TM, DEM and GIS data for landcover classification in support of grizzly bear habitat mapping. *International Journal of Remote Sensing* 23(21): 4633-4652.

Frazier, P. S., and Page, K., J., 2000. Water body detection and delineation with Landsat TM data. *Photogrammetric Engineering and Remote Sensing* 66(12): 1461-1468.

Fung, T., 1990. An Assessment of TM Imagery for Land-cover Ahange Detection. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 681-684

Ganas, A., Lagios, E., and Tzannetos, N., 2002. An investigation into the spatial accuracy of the IKONOS 2 ortho-imagery within an urban environment. *International Journal of Remote Sensing*, 23(17): 3513-3519.

Giles, P. T., 2001. Remote sensing and cast shadows in mountainous terrain, *Photogrammetric Engineering and Remote Sensing* 67(7): 833-840.

Gargantini, I., 1982. Detection of connectivity for regions represented by linear quadtrees. *Computer and Mathematics with Application* 8(4).

Godard, M., Gray, J., and Poitevin, J., 1990. The Relative Merits of Spot HRV and Landsat TM Images for Fourest Cover Change Detection in forillon National Park, Quebec, Canada. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 745-746.

Gomes J. and Velho, L., 1997. *Image Processing for Computer Graphics*. Springer-Verlag, New York.

Goodchild, M. F. 1980. Factals and the accuracy of geographical measures. *Mathematical Geology* 12(2): 85-98.

Goodchild, M. F., and Klinkenberg, B., 1993. Statistics of Channel Networks on Fractional Brownian Surfaces, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. Pp122-141

Goodchild, M. F. and Mark, D. M., 1987. The fractal nature of geographic phenomena. *Annals of the Association of American Geographers* 22(2): 265-278.

Gray, R. M., 1992. *Image Compression and Tree-Structured Vector Quantization*, in Image and Text Processing, edited by Storer, J. A., Kluwer Academic, Boston, pp 3-33.

Gupta, S. and Gersho, A., 1992. Feature Predictive Vector Quantization of Multispectral Images. *IEEE Transactions on Geoscience and Remote Sensing* 30(3): 491-501.

Hastings, H. M., and Sugihara, G., 1993. *Fractals: A User's Guide for the Natural Sciences*. Oxford Science Publications, New York.

He, G., 1988. *Fractal Model and Its Applications in Texture Recognition and Image Classification*. M.Sc. thesis, Institute of Remote Sensing Application, Chinese Academy of Sciences.

Heijmans, H. J. A. M., 1990. The Algebraic Basis of Mathematical Morphology 1. Dilations and Erosions. *CVGIP* 50(1) 245-295.

Hoffman R. N. and Johnson D. W., 1994. Application of EOFs to Multispectral Imagery: Data Compression and Noise Detection for AVIRIS. *IEEE Transactions on Geoscience and Remote Sensing* 32(1): 25-34.

Hord, R. M., 1986. *Remote Sensing Methods and Applications*. John Wiley & Sons, New York

Intergraph, 2000. *Take the GeoMedia 4.0 Challenge*. Intergraph Inc.

Jacquin, A. E., 1992. Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations. *IEEE Transaction on Image Processing*, 1(1): 18-30.

Jacquin, A. E., 1993. Fractal Image Coding: A Review. *Proceedings of the IEEE*, 81(10): 1451-1465.

Jahne, B., 1993. *Digital Image Processing: Concepts, Algorithms and Scientific Applications*. Springer – Verlag, New York.

Jain, A. K., 1981. Image Data Compression: A Review. *Proceedings of the IEEE*, 69(3): 349-389.

Jakubanskas, M. E., 1996. Thematic Mapper Characterization of Lodgepole Pine Seral Stages in Yellowstone National Park, USA. *Remote Sensing of Environment* 56(2): 118-132.

James, P. E. and Martin, G. J., 1981. *All Possible Worlds: A History of Geographical Ideas*. John Wiley & Sons, New York.

Jensen, J. R., 1996. *Introductory Digital Image Processing: a remote sensing perspective*. Prentice Hall, New Jersey.

Ji., C. Y., 2000. Land-use classification of remotely sensed data using Kohonen self-organizing feature mpa neural networks. *Photogrammetric Engineering and Remote Sensing* 66(12): 1451-1460.

Kettles, I. M., Rencz, A. N., and Bauke, S. D., 2000. Integrating Landsat, geologic and airborne Gamma ray data as an aid to surficial geology mapping and mineral exploration in the maniouwadge area, Ontario. *Photogrammetric Engineering and Remote Sensing* 66(4): 437-445.

Khorram, S., Brockhaus, J. A., Bruck, R. I., and Campbell, M. V., 1990. Modeling and Multitemporal Evaluation of Forest Decline with Landsat TM Digital Data. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 746-748.

Kniffen, F.b., 1988. *Louisiana, its land and people*. Louisiana State University Press, Baton Rouge.

Knipe, J. and Li, X., 1998. *On the Reconstruction of Quadtree Data*. *IEEE Transactions on Image Processing* 7(12): 1653-1660.

Kousias, N., Karteris, M., and Chuvico, E., 2000. The use of intensity–hue-saturation transformation of Landsat-t Thematic Mapper data for urned land mapping. *Photogrammetric Engineering and Remote Sensing* 66(7): 829-840.

Kstrand, S. E., 1994. Assessment of Forest Damage with Landsat TM: Correction for Varying Forest Stand Characteristics, *Remote Sensing of Environment* 47(3): 291-300.

Lagendijk, R. L. and Biemond, J., 1992. *Iterative Identification and Restoration of Images*. Kluwer Academic Publishers, Boston.

Lam, N. S. -N., 1990. Description and measurement of Landsat-TM images using fractals. *Photogrammetric Engineering and Remote Sensing* 56(2): 187-195.

Lam, N. S. -N. and Qiu, H.-L., 1992. *The fractal nature of the Louisiana coastline*. In Geographical Snapshots of North America, ed. D.G.Janelle, pp.270-274. New York: The Guilford Press.

Lam, N. S. -N. and Quattrochi, D. A., 1992. On the issues of scale, resolution, and fractal analysis in the mapping sciences. *The Professional Geographer* 44(1): 88-98.

Lam, N. S. -N., and De Cola, L., ed., 1993. *Fractals in Geography*, PTR Prentice-Hall, Engle wood Cliffs, NJ.

Lam, N. S. -N., Qiu, H. –L., Zhao, R., and Jiang, N., 1993. Afractal Analysis of Cancer Mortality Patterns in China, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp 247-262.

Lam, N. S.-N., Qiu, H.-L., Quattrochi, D. A., and Emerson, C. W., 2002, An Evaluation of Fractal Methods for Characterizing Image Complexity. *Cartography and Geographic Information Science*, 29(1): 25-35

Lammi, J. and Sarjakoski, T., 1995. Image Compression by the JPEG Algorithm. *Photogrammetric Engineering and Remote Sensing* 61(10): 1261-1266.

Lavallee, D., Lovejoy, S., Schertzer, D., and Ladoy, P., 1993. Nonlinear Variability of Landscape Topography: Multifractal Analysis and Simulation, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp158-192.

Lewis, R., 1990. *Practical Digital Image Processing*. Ellis Horwood, New York.

Lavery, P., et al., 1993. Water Quality Monitoring in Estuarine Water Using the Landsat Thematic Mapper. *Remote Sensing of Environment* 46(3): 468-480.

Li, Z., Yuan, X., and Lam, K., 2002. Effects of JPEG Compression on the Accuracy of Photogrammeric Point Determination. *Photogrammetric Engineering and Remote Sensing* 68(8) 847-853.

Lidov, L., Miller, R., Wormer, D. M., and Tilley, K. A., 2000. Understanding the Future of Commercial Remote Sensing. *Photogrammetric Engineering and Remote Sensing* 66(1): 5-16.

Lillesand, T. M., and Kiefer, R. W., 1999. *Remote Sensing and Image Interpretation, 4th edition*. John Wiley & Sons Inc., New York.

Lymburner, L., Beggs, P. J., and Jacobson, C. R., 2000. Estimation of Canopy-aveage surface- specific leaf area using Landsat TM data. *Photogrammetric Engineering and Remote Sensing* 66(2): 183-192.

Lynch, T. J., 1985. *Data Compression Techniques and Applications*. Wiley, New York.

Mandelbrot, B. B., 1967. How Long is the Coast of Britain? Statistical Self-similarity and Fractional Dimension. *Science* 156: 636-638.

Mandelbrot, B. B., 1977. *Fractals, Form, Chance, and Dimension.* W.H.Freeman, San Francisco.

Mandelbrot, B. B., 1983. *The Fractal Geometry of Nature*. Freeman , New York,.

Martin, N. F. G., and England, J. W., 1981. *Mathematical Theory of Entropy*. Addison – Wesley Publishing Company, Reading, MA.

Memon, N. D., Sayood, K., and Magliveras, S. S., 1994. Lossless Compression of Multispectral Image Data. *IEEE Transactions on Geoscience and Remote Sensing* 32(2): 282-289.

Miano, J., 1999. *Compressed Image File Formats, JPEG, PNG, GIF, XBM, BMP*. , Addison-Wesley, New York.

Migual-Avanz J. S. and Biging, G. S., 1997. Comparison of Single-Stage and Multi-Stage Classification Approaches for Cover Type Mapping with TM and Spot Data. *Remote Sensing of Environment* 59(1): 92-104.

Miliaresis, G. C., and Argialas, D., P., 2000. Extraction and delineation of alluvial fans from digital elevation models and Landsat Thematic Mapper images. *Photogrammetric Engineering and Remote Sensing* 66(9): 1093-1102.

Moran, M. S., 1995. Reflectance Factor Retrieval from Landsat TM and Spot HRV Data for Bright and Dark Targets. *Remote Sensing of Environment* 52(3): 218-230.

Neville, P., Coward, R., I., Watson, R. P., Inglis, M., and Morain, Stan, 2000. The application of TM imagery and GIS data in the assessment of arid lands water and resources in West Texas. *Photogrammetric Engineering and Remote Sensing* 66(11): 1373-1382.

Nijim, Y. W., Steams, S. D., and Mikhael, W. B., 1996. Lossless Compression of Seismic Signal Using Differentiation. *IEEE Transactions on Geoscience and Remote Sensing* 34(1): 52-56.

Novak K. and Shahin, F. S., 1996. A Comparison of Two Image Compression Techniques for Softcopy Photogrammetry. *Photogrammetrical Engineering & Remote Sensing*, 62(6): 695-701.

Pei, S., Tseng, C., and Lin C., 1996. A Parallel Decoding Algorithm for IFS Codes without Transient Behavior. *IEEE Transactions on Image Processing*, 5(3): 411-415.

Pennehaker, W. B. and Mitchell, J. L., 1993. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York.

Phillips, J., 1993. Interpreting the Fractal Dimension of River Networks, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp 142-157.

Plotnick, R., and Prestegaard, K., 1993. Fractal Analysis of Geologic Time Series, in *Fractals in Geography*, edited by Lam, N. S. -N., and De Cola, L., PTR Prentice-Hall, Engle wood Cliffs, NJ. pp193-210.

Pratt, W. K., 1978. *Digital Image Processing*. John Wiley & Sons, New York.

Pugh, S. A., and Congalton, R. G., 2001. Applying spatial autocorrelation analysis to evaluate error in New England forest-cover-type maps derived from Landsat Thematic Mapper data. *Photogrammetric Engineering and Remote Sensing* 67(5): 613-620.

Oppenheimer, C., and Yirgu, G., 2002. Thermal imaging of an active lava lake: Erta ‘Ale Volcano, Ethiopia. *International Journal of Remote Sensing* 23(22): 4777-4782.

Quattrochi, D. A., Lam, N. S. -N, Qiu, H. L., and Zhao, W., 1997. Image Characterization and Modeling Systm(ICAMS): *A Geographic Information System for the Characterization and Modeling of Multiscale Remote Sensing Data*. In Scaling of Remote Sensing Data for GIS, edited by Quattrochi, D. and Goodchild, M., CRC/Lewis Publishers, Boca Raton, Florida.

Reafat, H. M., Xiao, Q., and Gauthier. D.A., 1991. An Extended Relational Database for Remotely Sensed Image Data Management within GIS. *IEEE Transactions on Geoscience and Remote Sensing* 29(4): 651-655.

Ricchetti, E., 2000. Multispectral satellite image and ancillary data integration for geological classification. *Photogrammetric Engineering and Remote Sensing* 66(4): 429-436.

Roger, R. E., and Arnold, J. F., 1994. Reversible Image Comression Bounded by Noise. *IEEE Transactions on Geoscience and Remote Sensing* 32(1): 19-24.

Roger, R. E., and Cavenor, M. C., 1996. Lossless Compression of AVIRIS Images. *IEEE Transactions on Image Processing* 5(5): 713-719.

Roman, S., 1997. *Introduction to Coding and Information Theory*. Springer, New York.

Russ, J. C., 1999. *The Image Processing Handbook* 2nd edition. CRC Press, Boca Raton, Fla.

Ryan, M. J. and Arnold, J. F., 1997. The lossless Compression of AVIRIS Images by Vector Quatization. *IEEE Transactions on Geoscience and Remote Sensing* 35(3): 546-550.

Sader, S. A., Ahl, D., and Lion W. -S., 1995. Accuracy of Landsat TM and GIS Rule-Based Methods for Forest Wetland Classification in Maine. *Remote Sensing of Environment* 53(3): 133-144.

Saha, S., 2002. Image Compression – from DCT to Wavelets: A Review. *ACM Crossroads Student Magazine* (ACM's First Electronic Publication).

Said, A. And Pearlman, W. A., 1996. An Image Multiresolution Representation for Lossless and Lossy Compression. *IEEE Transactions on Image Processing* 5(9): 1303-1310.

Saupe, D. and Hamzaoui, R., 1994. A Review of the Fractal Image Compression Literature. *Computer Graphics* 28(5): 268-276.

Sayood, K., 1996. *Introduction to Data Compression*. Morgan Kaufmann Publishers, Inc., San Francisco, California

Schowengerdt, R. A., 1997. *Remote Sensing Models and Methods for Image Processing*. Springer, New York.

Schott, J. R., 1991. *Remote Sensing — The Chain Approach*. Oxford University Press, New York.

Shaykh, O. K. A. and Merserean, R.M., 1998. Lossy Compression of Noisy Images. *IEEE Transactions on Image Processing* 7(12): 1641-1652.

Shelberg, M. C., Moellering, H., and Lam, N., 1982. Measuring the fractal dimensions of empirical cartographic curves. *Proceedings of the 5th International Symposium on Computer-assisted Cartography* (Auto-Carto 5): 481-490.

Shelberg, M. C., Lam, N. and Moellering, H., 1983. Measuring the fractal dimensions of surfaces. *Proceedings of the 6th International Symposium on Computer-Assisted Cartography* (Auto-Carto 6), vol. 2, pp319-328.

Shen, S. S., 1989. *AnAdaptive DPCM Compression Scheme For Real-Time Applications*. in SPIE Vol. 1099 Advances in Image Compression and Automatic Target Recognition, pp124-133.

Smit, P. H., 2001. Maximizing information content of Landsat imagery for coastal zone applications. *Photogrammetric Engineering and Remote Sensing* 67(7): 769-772.

Solberg, A. S., Jain, A. K., and Taxt, T., 1994. Multi-source Classification of Remotely Sensed Data: Fusion of Landsat TM and SAR Images. *IEEE Transactions on Geoscience and Remote Sensing* 32(4): 768-778.

Star J. L., Estes, J. E., and McGwire, K. C., ed., 1997. *Integration of Geographic Information Systems and Remote Sensing*. Cambridge University Press, New York.

Stern, A., Doraiswamy, P., and Cook, P., 2001. Spring Wheat Classification in an AVHRR image by Signature Extension from a Landsat TM Classified Image. *Photogrammetric Engineering and Remote Sensing* (67)2: 207-212.

Sternberg, S. R., 1986. Grayscale Morphology. *Computer Vision Graphics and Image Processing* Vol.35: 333-355.

Stored, J. A., ed., 1992. *Image and Text Compression*. Klunwer Academic Publishers, Boston.

Thao., N. T. and Vetterli, M., 1998. Set Theoretic Compression with an Application to Image Coding. *IEEE Transactions on Image Processing* 7(7): 1051-1056.

Thenkabai, P. S., Nolte, C., and Lyon, J. G., 2000. Remote sensing and GIS modeling for selecting of a benchmark research area in the inland valley agroecosystems of West and Central Africa. *Photogrammetric Engineering and Remote Sensing* 66(6): 755-768.

Tnagestani, M. H., and Moore, F., 2002. Porphyry copper alteration mapping at the Meiduk area, Iran. *International Journal of Remote Sensing* 23(22): 4815-4826.

Ton, J., Sticklen., J., and Jain, A. K., 1991. Knowledge-Based Segmentation of Landsat Images. *IEEE Transactions on Geoscience and Remote Sensing* 29(2): 222-232.

Vaisey, J., Gersho, 1992. A., Image Compression with Variable Block Size Segmentation. *IEEE Transactions on Signal Processing* 40(8): 2040-2060.

Vogelmann, J., E., Howard, S. M., Yang, L., Larson, C. r., Wylie, B. K., and Driel, N., V., 2001. Completion of the 1990s National Land Cover Data Set for the conterminous United States from Landsat Thematic Mapper data and Ancillary data Cources. *Photogrammetric Engineering and Remote Sensing* 67(6): 650-684.

Wang, G., Gertner, G., Xiao, X., Wente, S., and Anderson, A., B., 2001. Appropriate plot size and spatial resolution for mapping multiple vegetation types,.*Photogrammetric Engineering and Remote Sensing* 67(5): 575-584.

Wang, J, and Howarth, P. J., 1990. Use of the Hough Transform in Automated Linear Detection. *IEEE Transactions on Geoscience and Remote Sensing* 28(4): 561-567.

Wang, J., Zhang, K., and Tang, S., 1995. Spectral and Spatial Decorrelation of Landsat TM Data for Lossless Compression. *IEEE Transactions on Geoscience and Remote Sensing* 33(5): 1277-1285.

Wang, Y., Colby, J. D., and Mulcahy, A., 2002. An efficient method for mapping flood extent in a coastal floodplain using Lansat TM and DEM data. *International Journal of Remote Sensing* 23(18): 3681-3696.

Wayman, J. P., Wynne, R. H., Scrivani, J. A., and Reams, G. A., 2001. Landsat TM-based forest area estimation using iterative guided spectral class rejection. *Photogrammetric Engineering and Remote Sensing* 67(10): 1155-1166.

Webb, E., L., Evangelista, M. A., and Robinson, J. A., 2000. digital land-use classification using space-shuttle-acquired orbital photographs: a quantitative comparison with Landsat TM imagery of a coastal environment, Chanthaburi, Thailand. *Photogrammetric Engineering and Remote Sensing* 66(12): 1439-1450.

Weigel, S. J., 1996. *Scale, Resolution and Resampling: representation and analysis of remotely sensed landscapes across scale in Geographical Information Systems*. Ph.D. Dissertation, Department of Geography & Anthropology, Louisiana State University.

Wohlberg, B. and Jager, G., 1999. A review of the fractal image coding literature. *IEEE Transactions on Image Processing* 8(12): 1716-1729.

Woodcock, C. E., 1994. Mapping Forest Vegetation Using Landsat TM Imagery and a Canopy Reflectance Model. *Remote Sensing of Environment* 50(3): 436-451.

Xiao, K., 1991. The Image's Quadtree Structures, Their Research, Application and Development. *Remote Sensing Technology and Application* 6(1): 18-25.

Xiao, K., 1992. The algorithm Converting Raster to Quadtree. *The Proceedings of 2nd International Conference on Automatics, Robotics, and computer vision*, September 1992, Singapore.

Xiao, K., 1993. The Algorithm Converting 2DRE Quadtree to Raster. *Enviromental Remote Sensing China* 8(1): 30-36.

Xiao, X., Boles, S., and Frolking, S., 2002. Landscape-scale characerization of cropland in China using Vegetation and Landsat TM images. *International Journal of Remote Sensing* 23(18): 3595-3594.

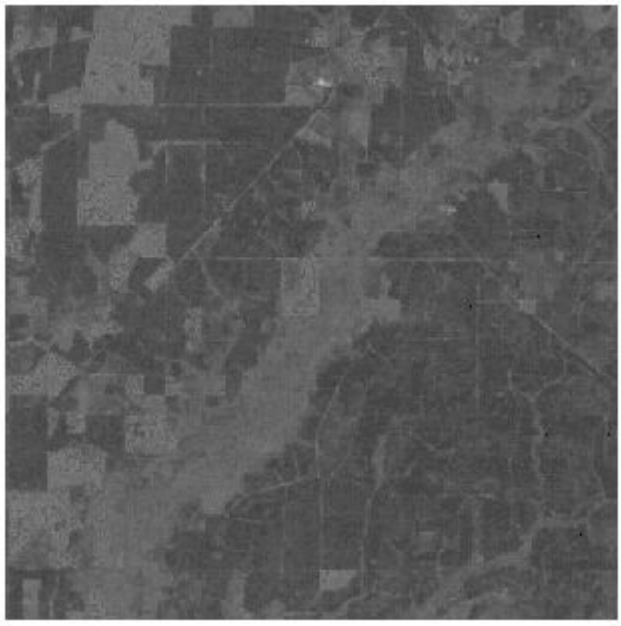
Xu, T., Moore, I. D., and Gallant, J. C., 1993. Fractals Fractal Dimensions and Landscapes – A Review. *Geomorphology*, Vol. 8: 245-262.

Zhang, Y., 2000. A method for continuous extraction of multispectrally classified urban rivers. *Photogrammetric Engineering and Remote Sensing* 66(8): 991-1000.

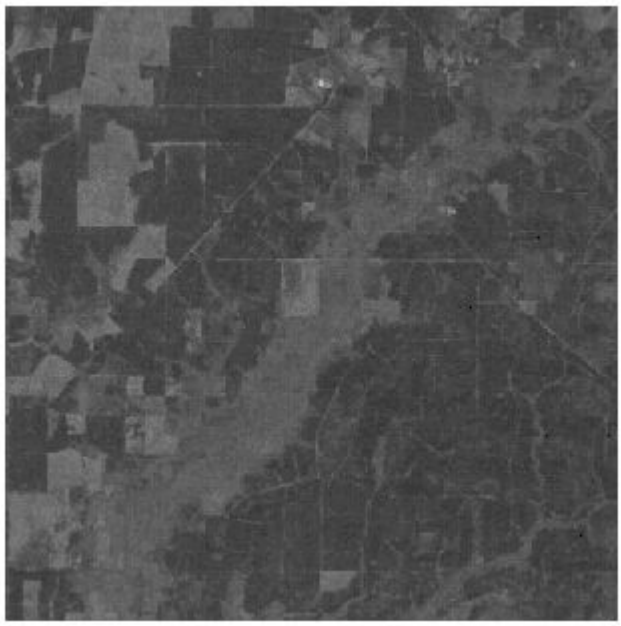
Zhao, W., 2001. *Multiscale Analysis for Characterization of Remotely Sensed Image*. Ph.D. Dissertation, Department of Geography & Anthropology, Louisiana State University.

APPENDIX A

COMPARISON OF ORIGINAL AND RESTORED IMAGES



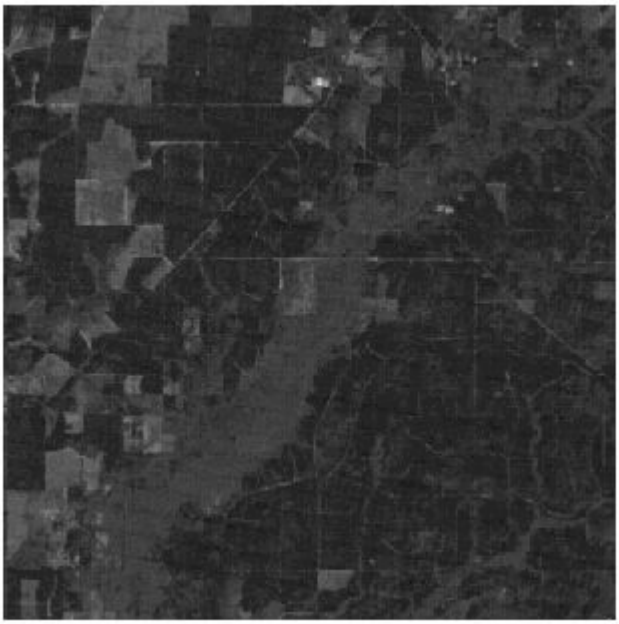
(a)



(b)

Figure A.1 Original and Restored images of Area-1, Band-1

(a)



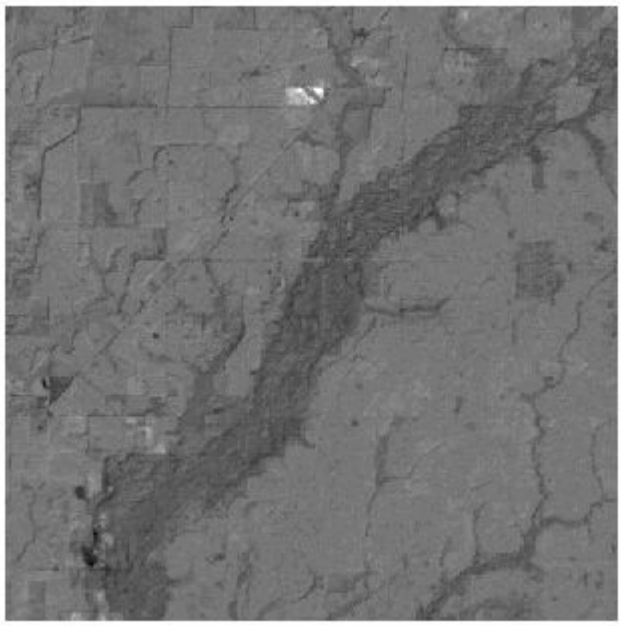
(b)

Figure A.2 Original and Restored images of Area-1, Band-2

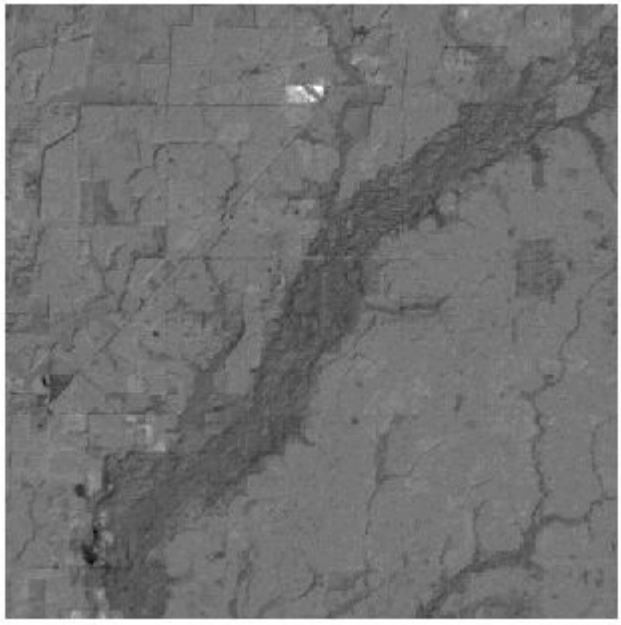
(a)

(b)

Figure A.3 Original and Restored images of Area-1, Band-3

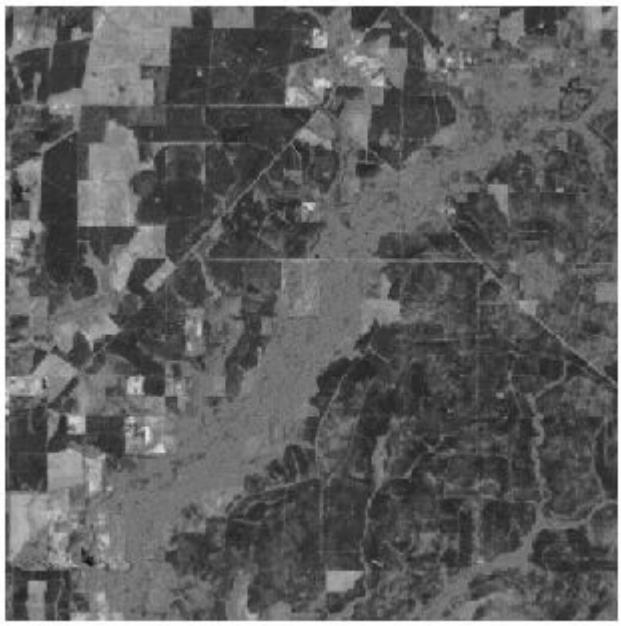


(a)

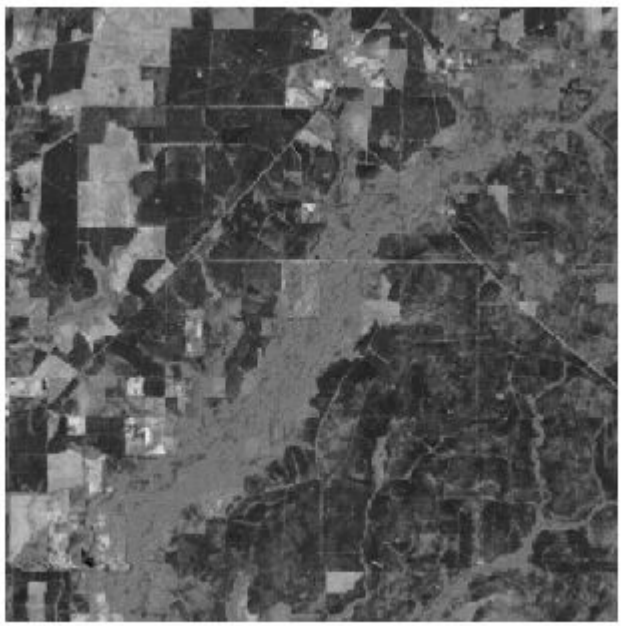


(b)

Figure A.4 Original and Restored images of Area-1, Band-4

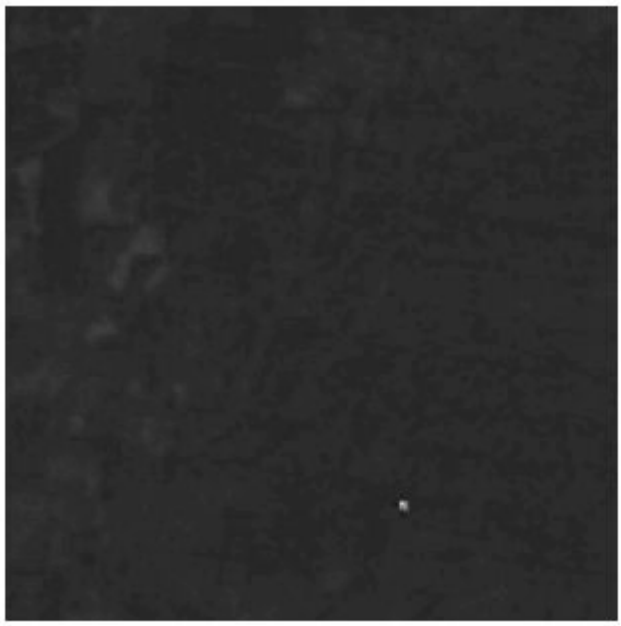


(a)

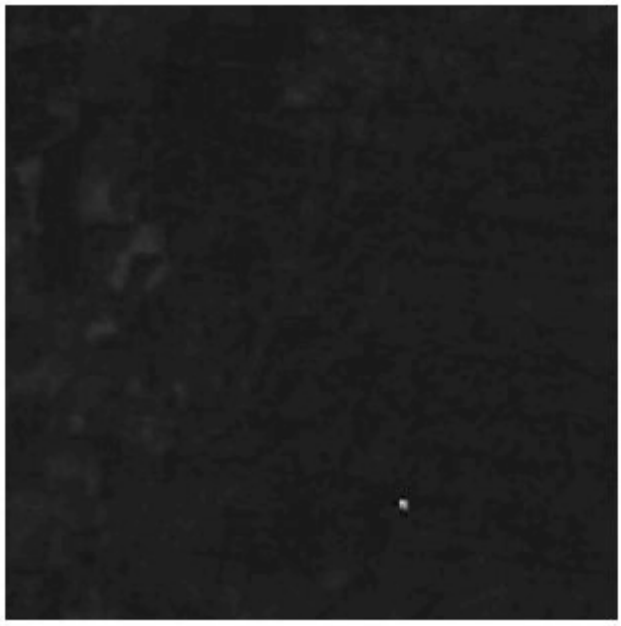


(b)

Figure A.5 Original and Restored images of Area-1, Band-5



(a)



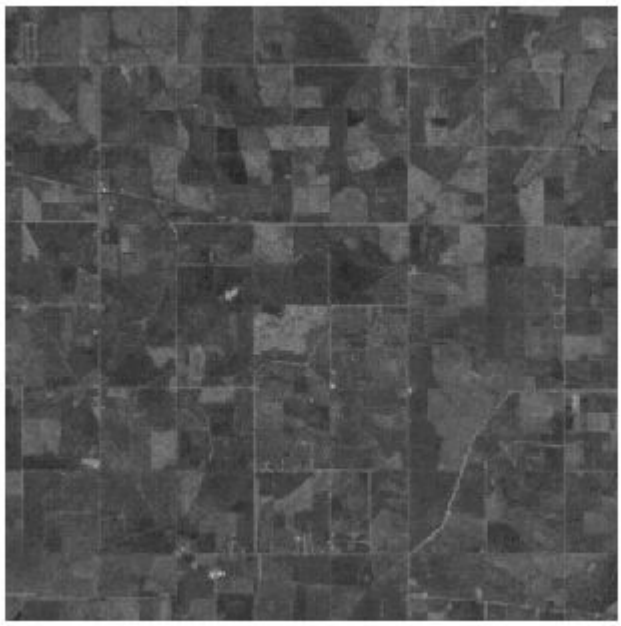
(b)

Figure A.6 Original and Restored images of Area-1, Band-6

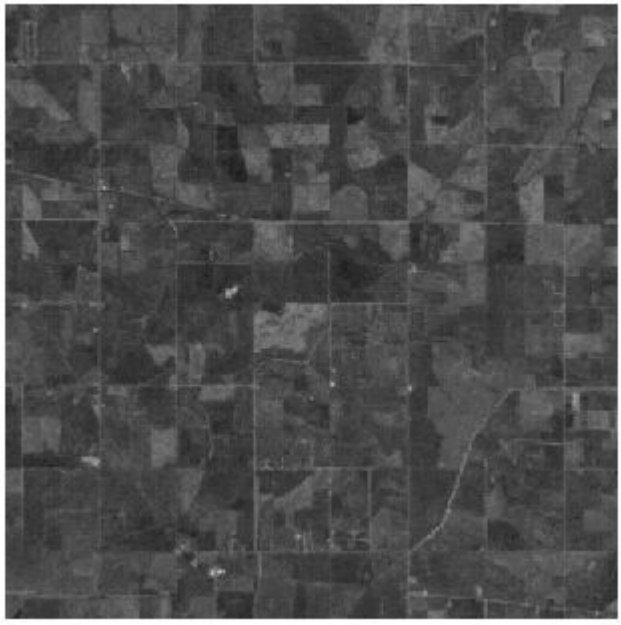
(a)

(b)

Figure A.7 Original and Restored images of Area-1, Band-7

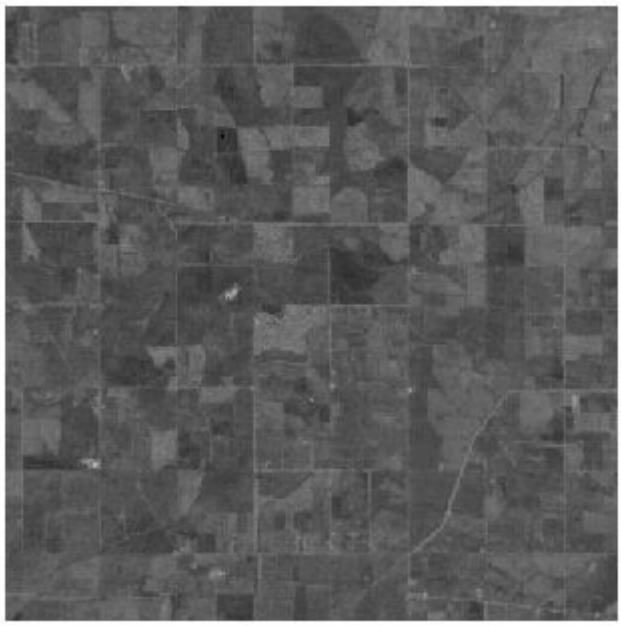


(a)

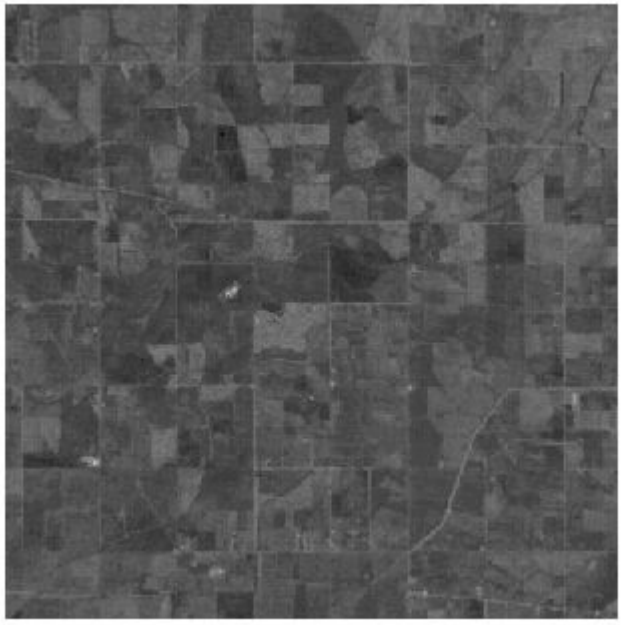


(b)

Figure A.8 Original and Restored images of Area-2, Band-1

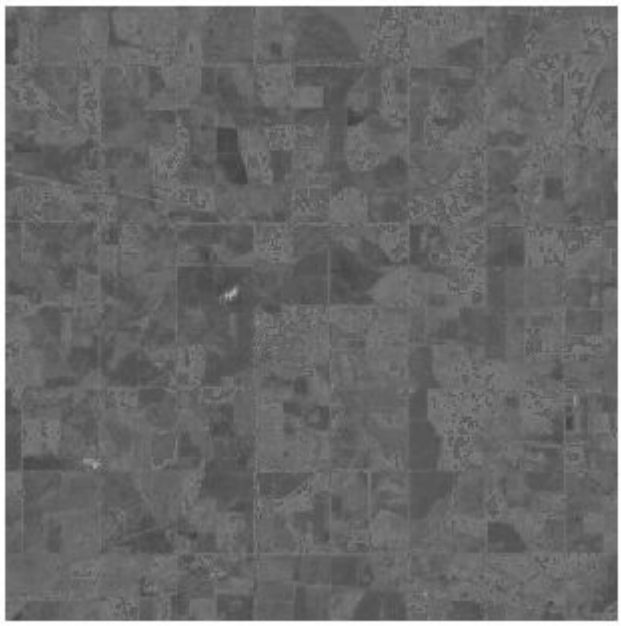


(a)

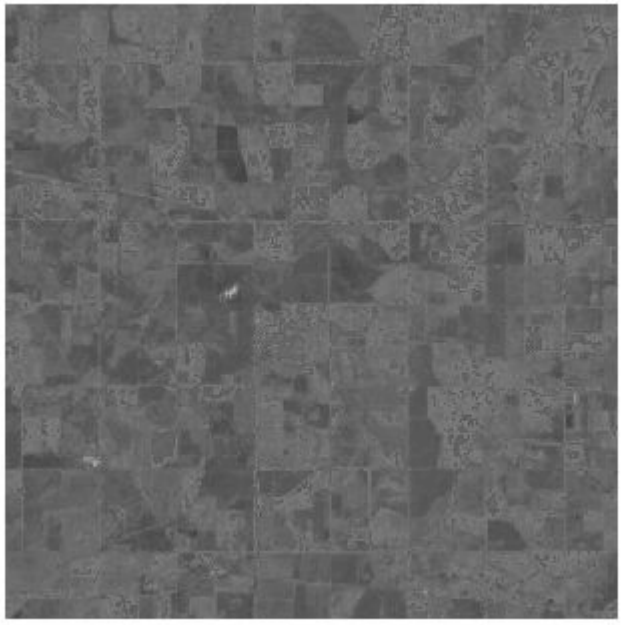


(b)

Figure A.9 Original and Restored images of Area-2, Band-2

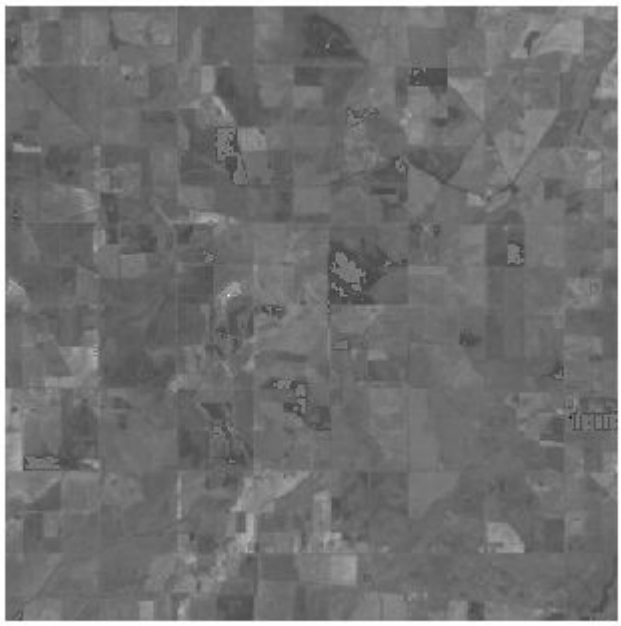


(a)

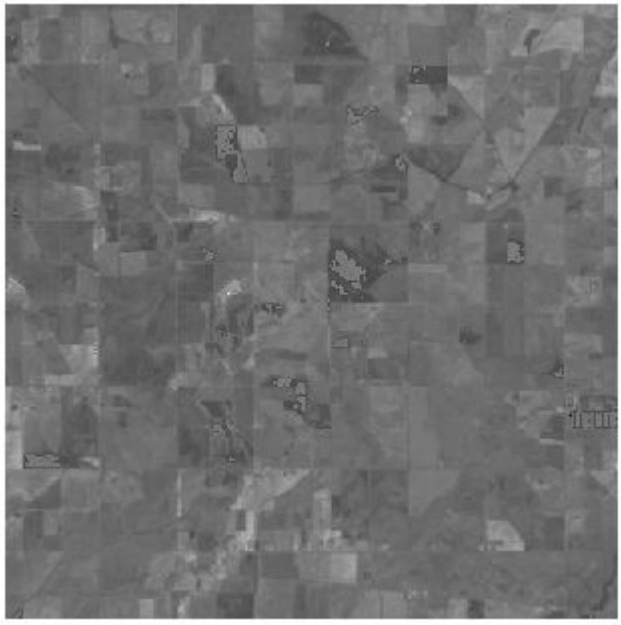


(b)

Figure A.10 Original and Restored images of Area-2, Band-3

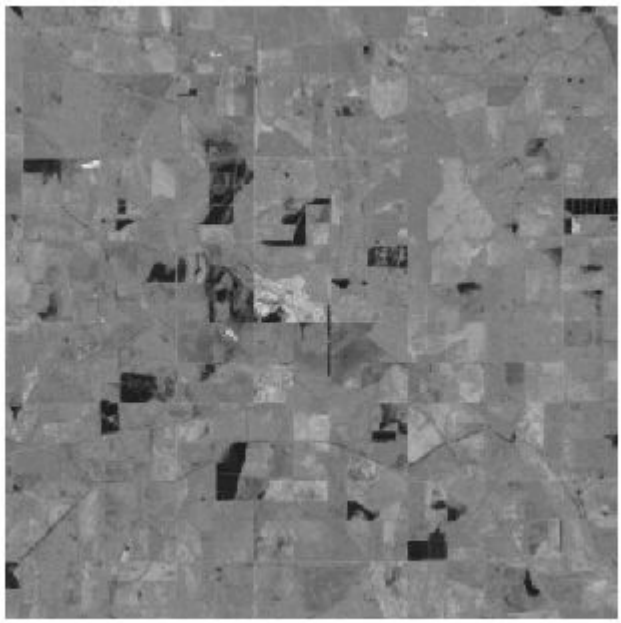


(a)

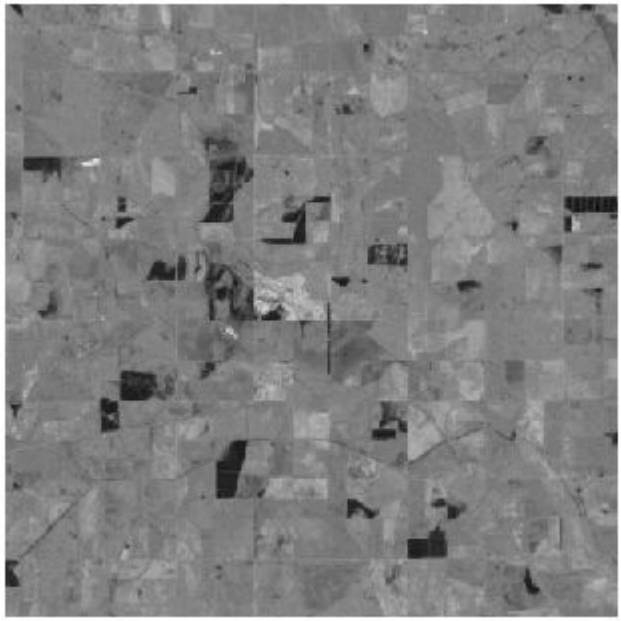


(b)

Figure A.11 Original and Restored images of Area-2, Band-4

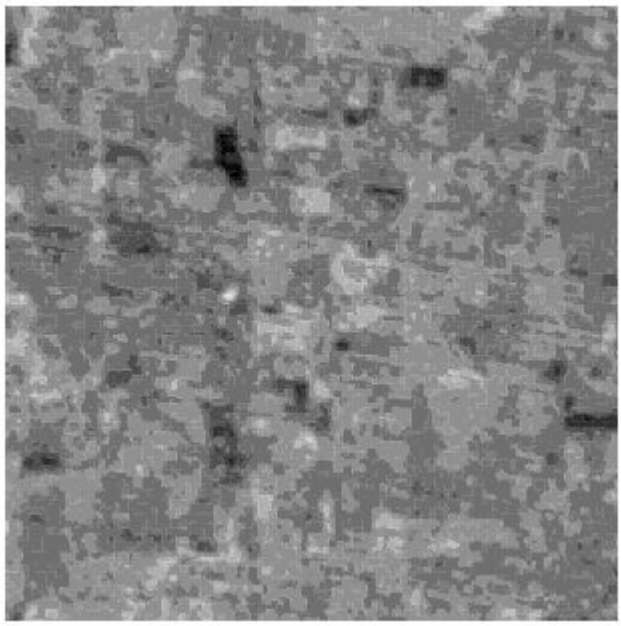


(a)

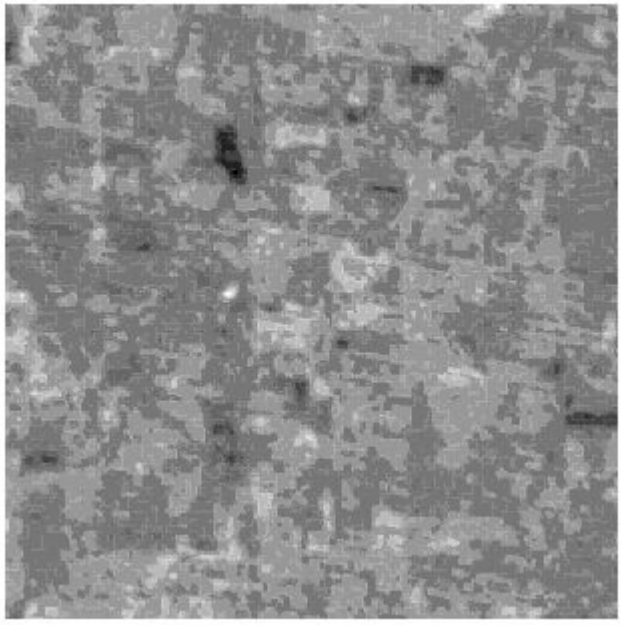


(b)

Figure A.12 Original and Restored images of Area-2, Band-5

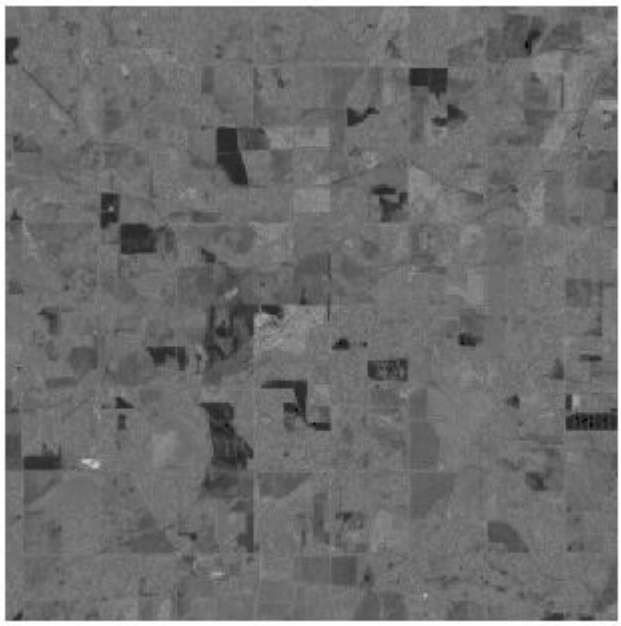


(a)

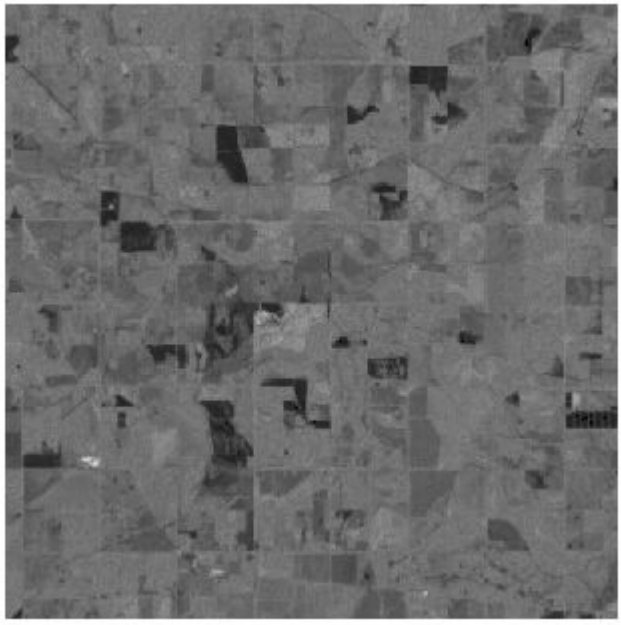


(b)

Figure A.13 Original and Restored images of Area-2, Band-6



(a)



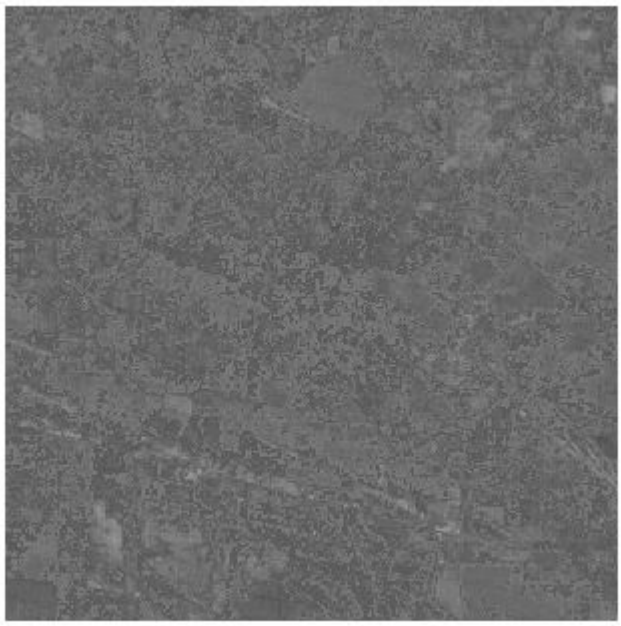
(b)

Figure A.14 Original and Restored images of Area-2, Band-7

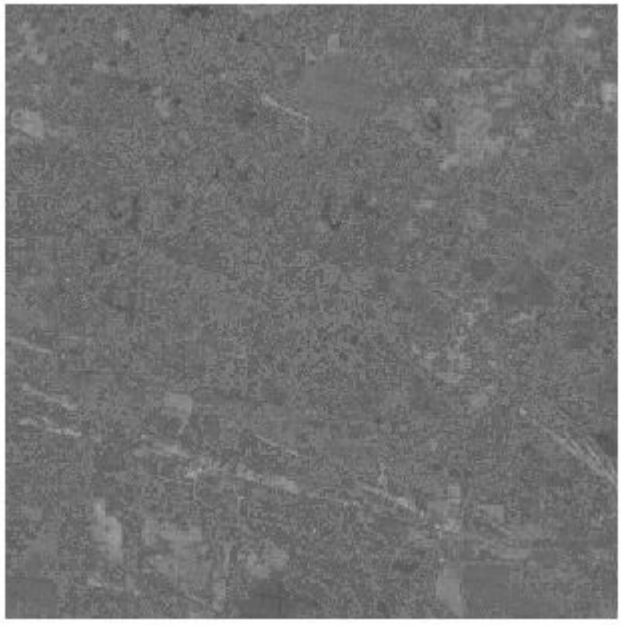
(a)

(b)

Figure A.15 Original and Restored images of Area-3, Band-1

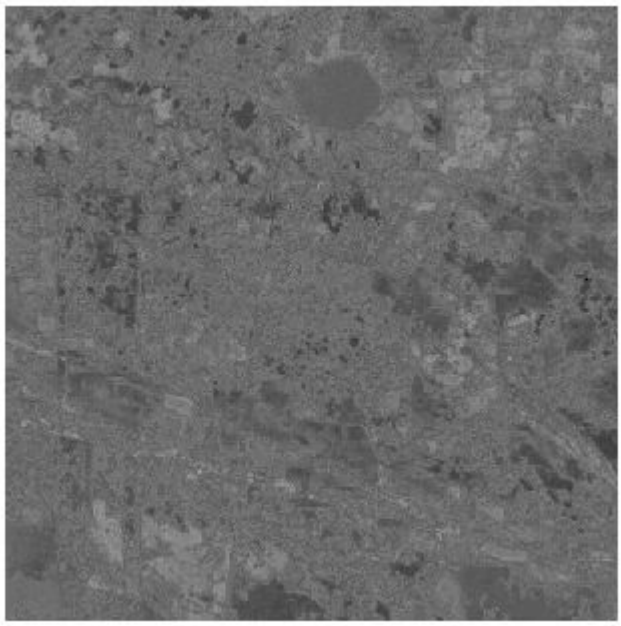


(a)

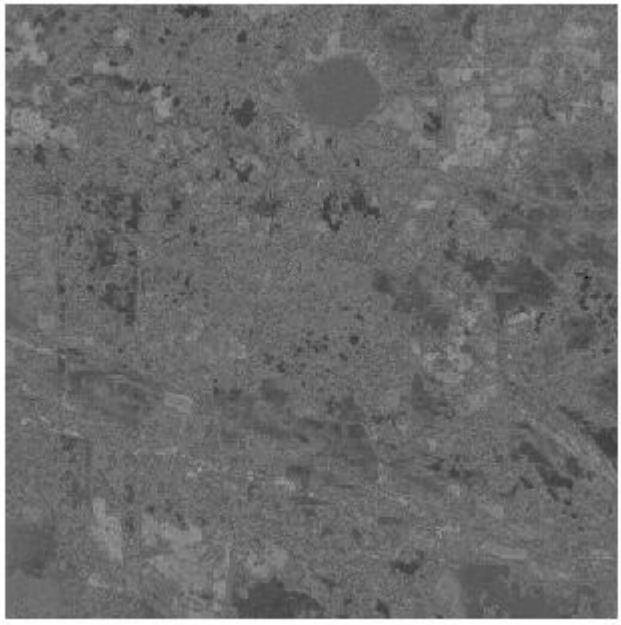


(b)

Figure A.16 Original and Restored images of Area-3, Band-2

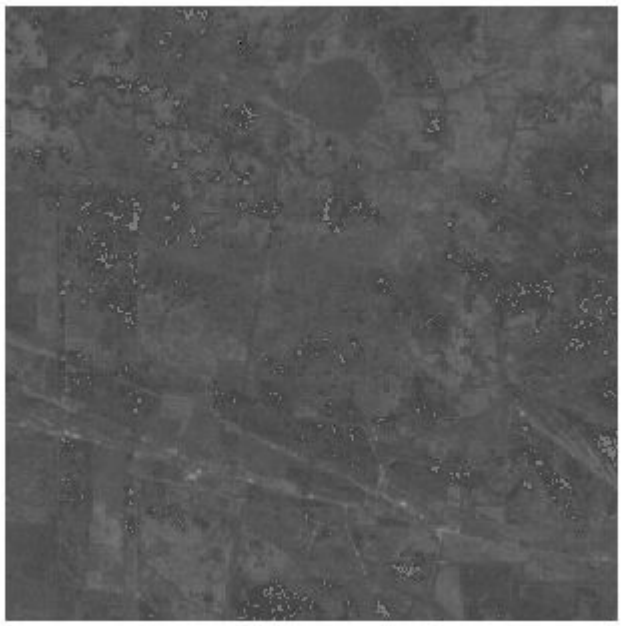


(a)



(b)

Figure A.17 Original and Restored images of Area-3, Band-3



(a)

(b)

Figure A.18 Original and Restored images of Area-3, Band-4

(a)

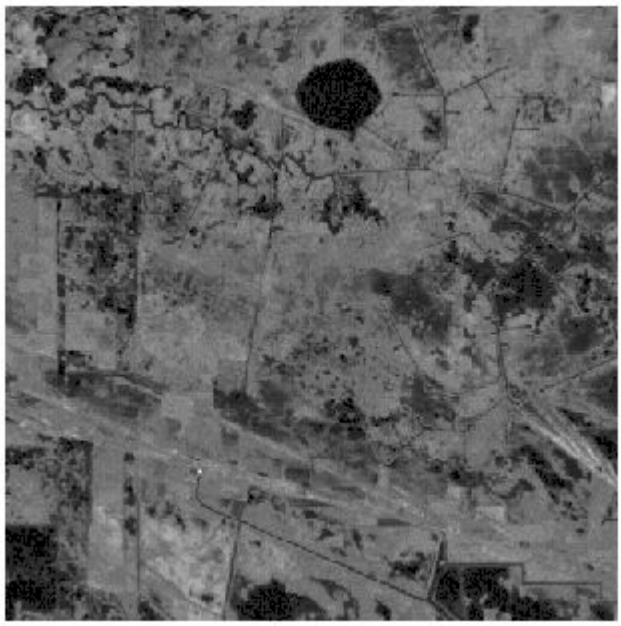
(b)

Figure A.19 Original and Restored images of Area-3, Band-5

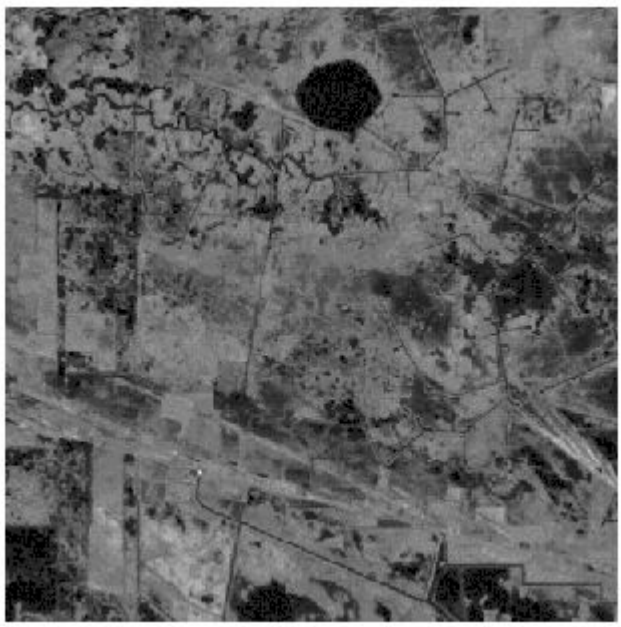
(a)

(b)

Figure A.20 Original and Restored images of Area-3, Band-6

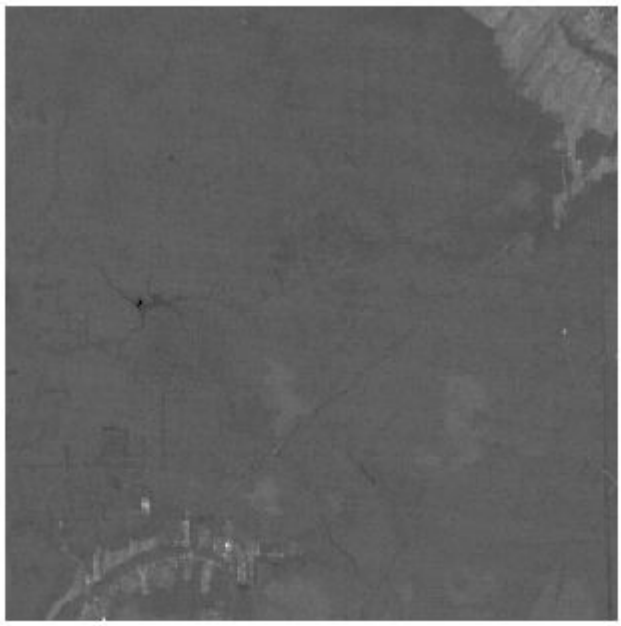


(a)

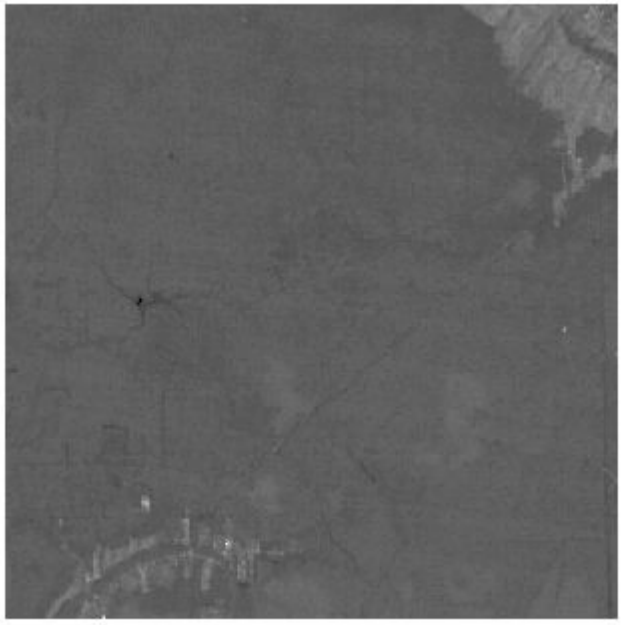


(b)

Figure A.21 Original and Restored images of Area-3, Band-7

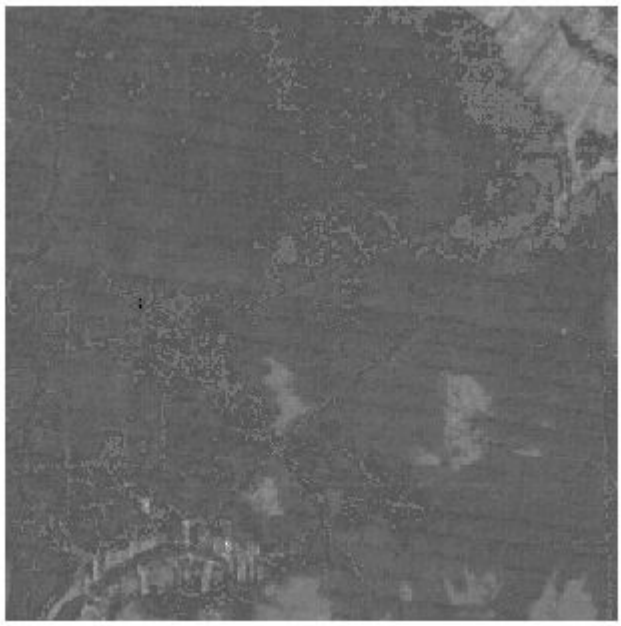


(a)

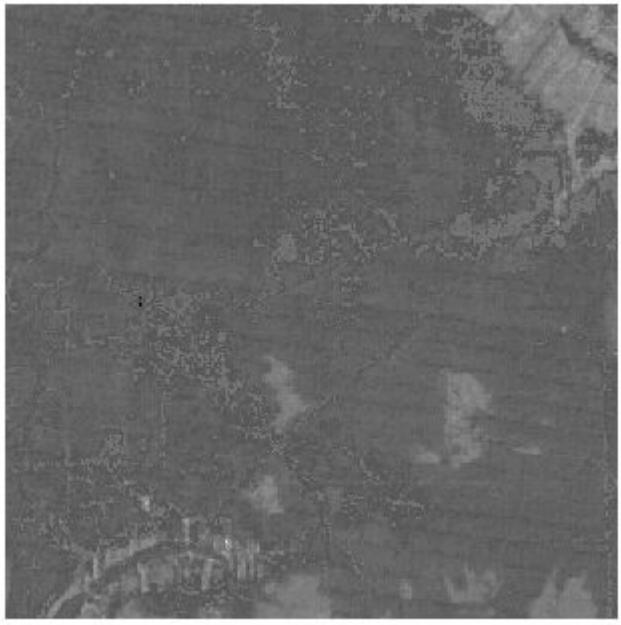


(b)

Figure A.22 Original and Restored images of Area-6, Band-1

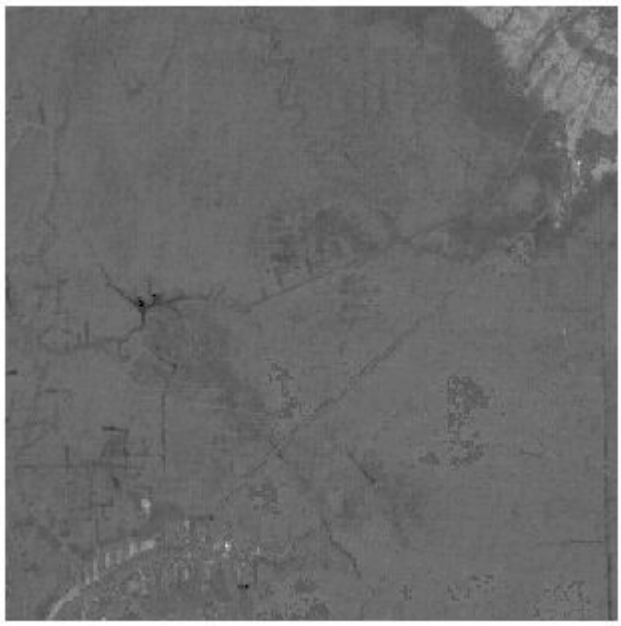


(a)

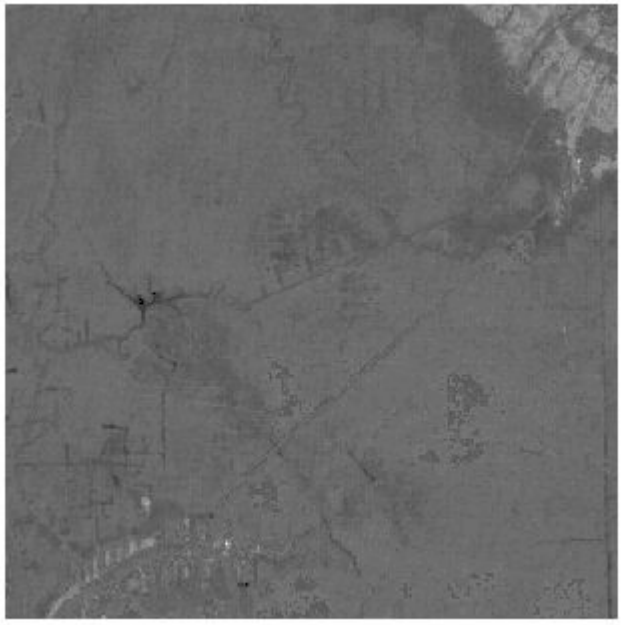


(b)

Figure A.23 Original and Restored images of Area-6, Band-2

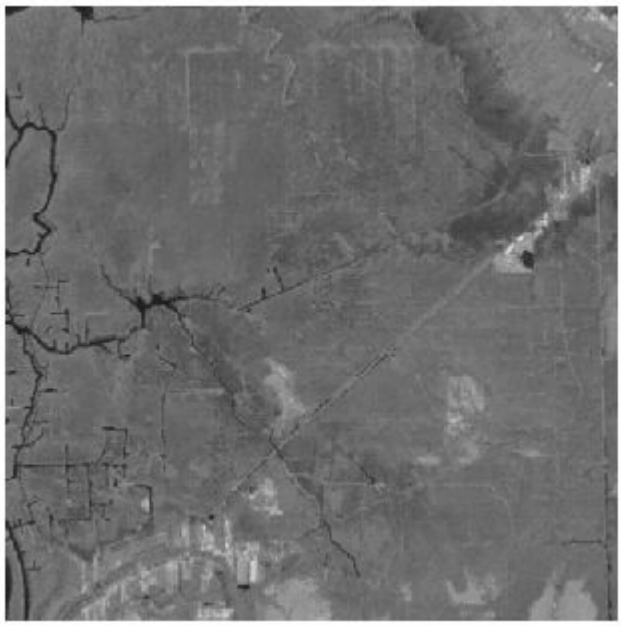


(a)

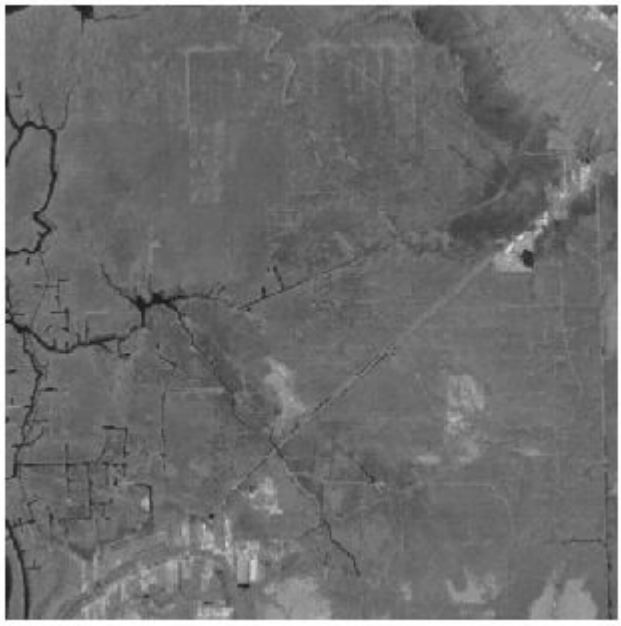


(b)

Figure A.24 Original and Restored images of Area-6, Band-3

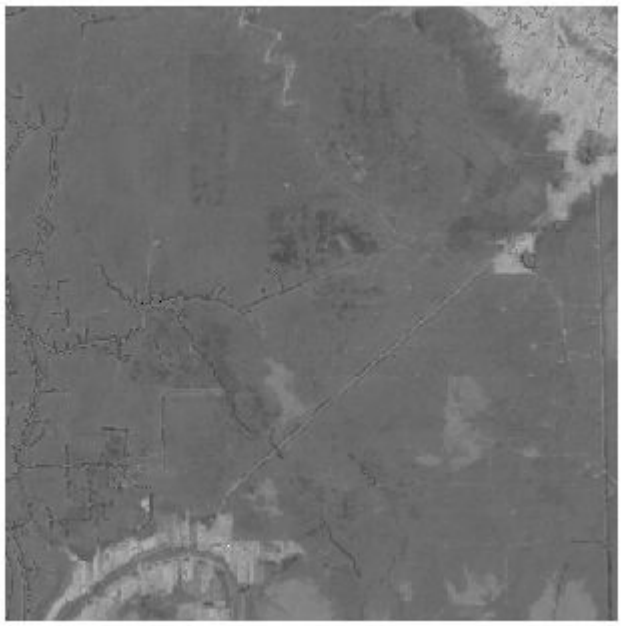


(a)

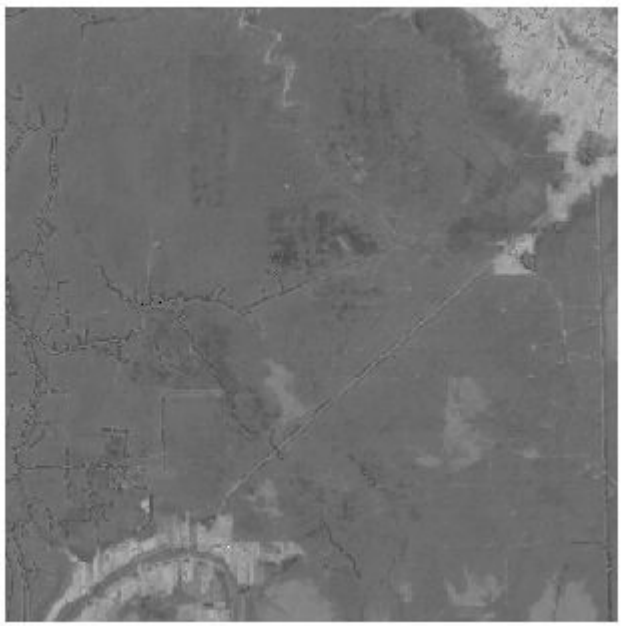


(b)

Figure A.25 Original and Restored images of Area-6, Band-4

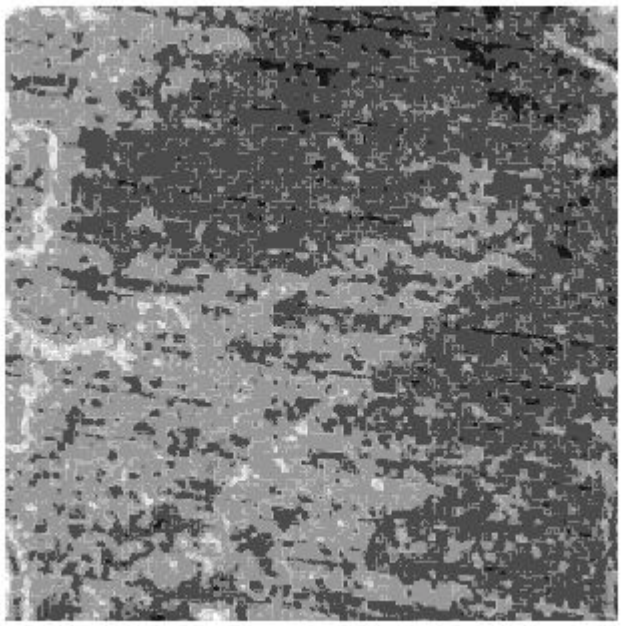


(a)

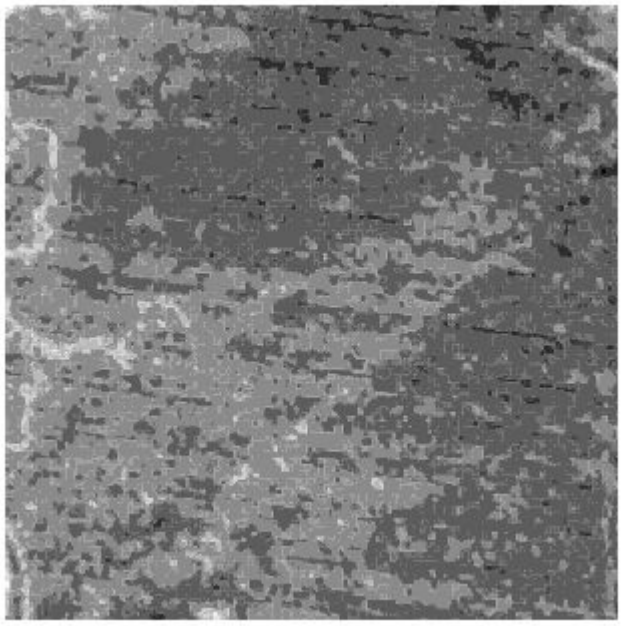


(b)

Figure A.26 Original and Restored images of Area-6, Band-5

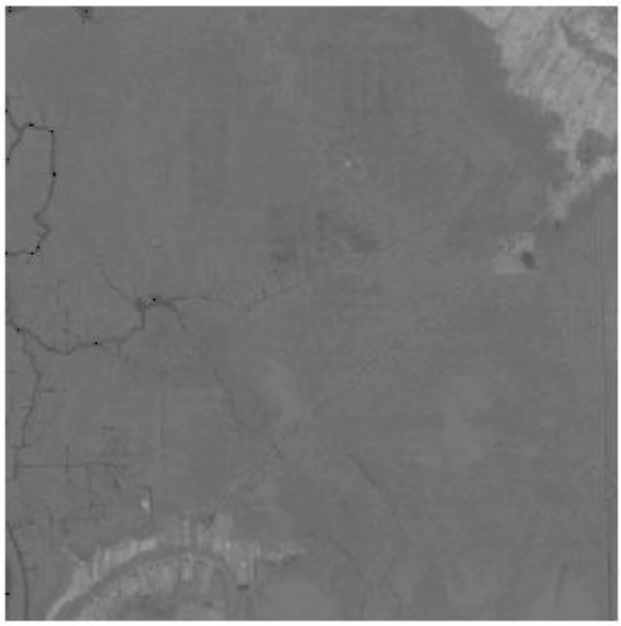


(b)

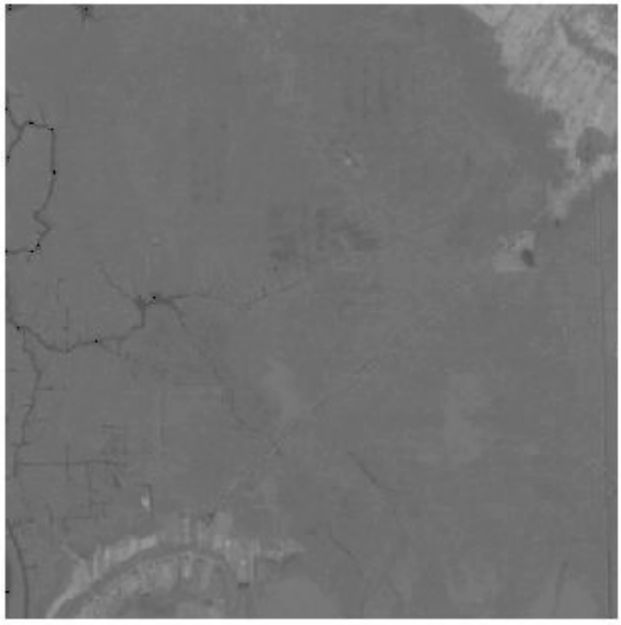


(b)

Figure A.27 Original and Restored images of Area-6, Band-6

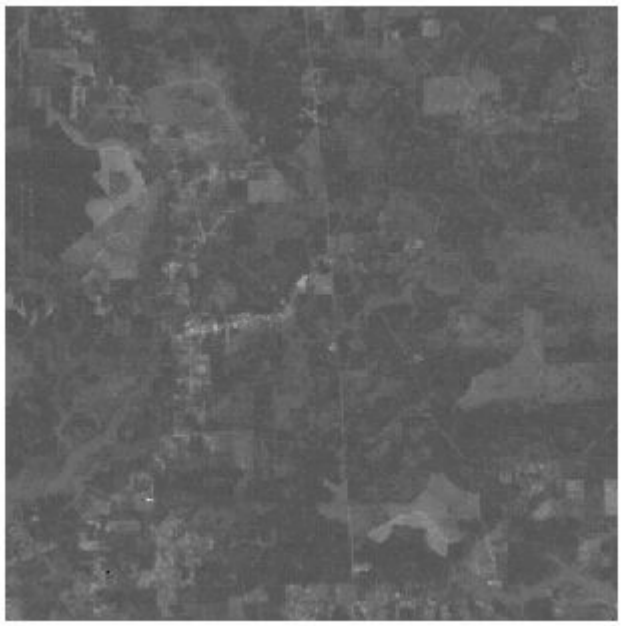


(a)

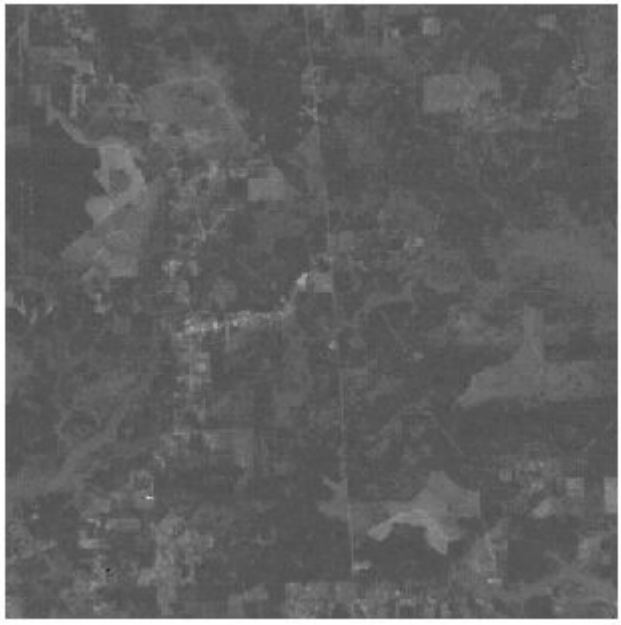


(b)

Figure A.28 Original and Restored images of Area-6, Band-7

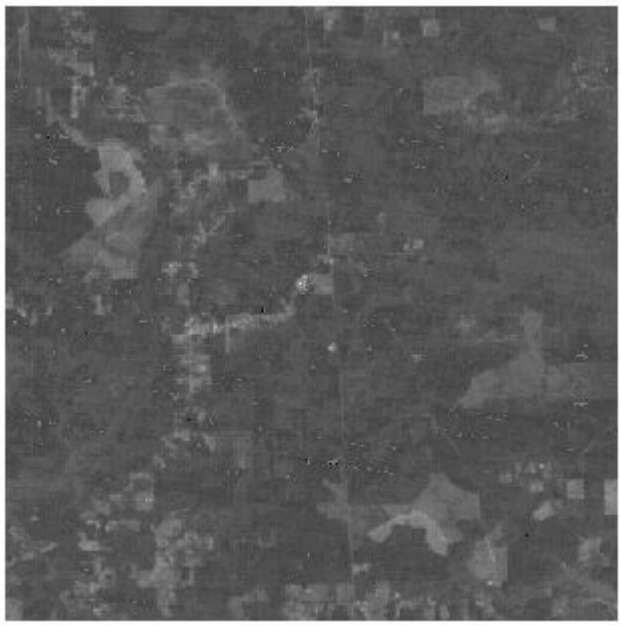


(a)

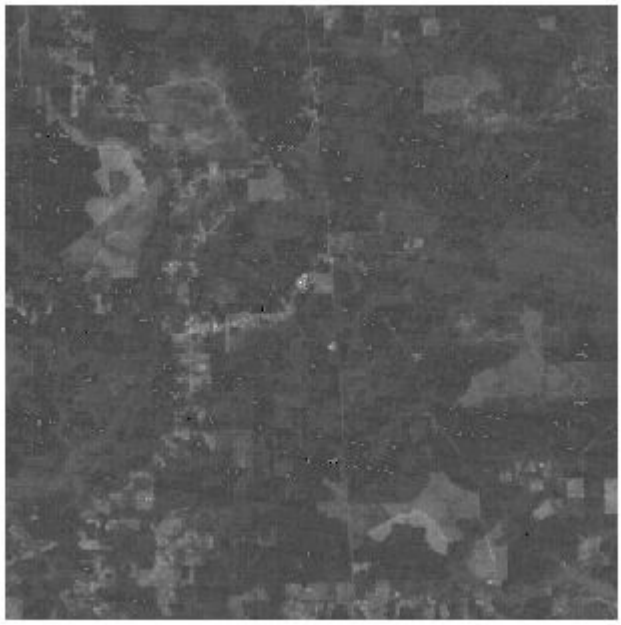


(b)

Figure A.29 Original and Restored images of Area-7, Band-1

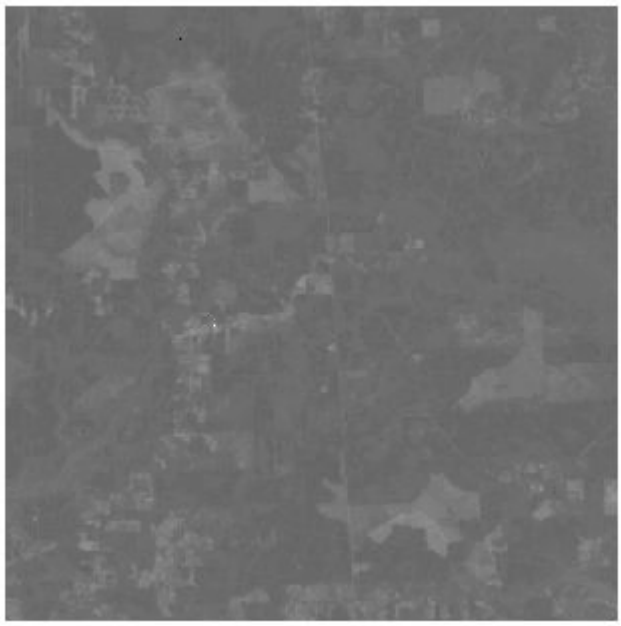


(a)

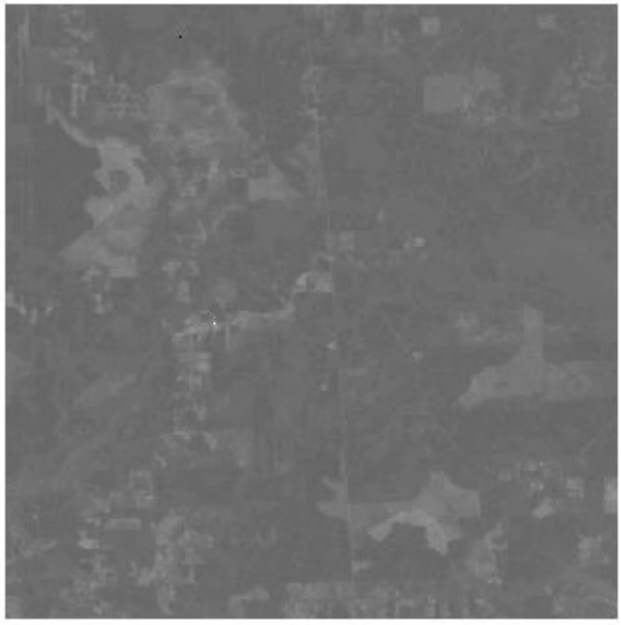


(b)

Figure A.30 Original and Restored images of Area-7, Band-2

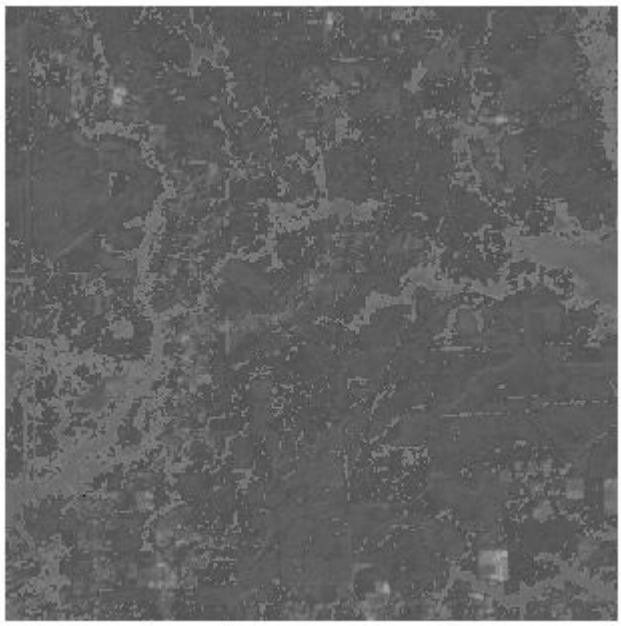


(a)

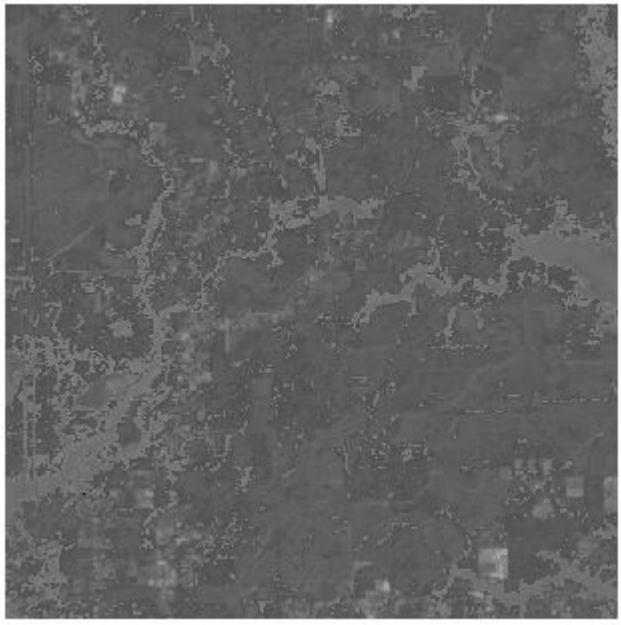


(b)

Figure A.31 Original and Restored images of Area-7, Band-3

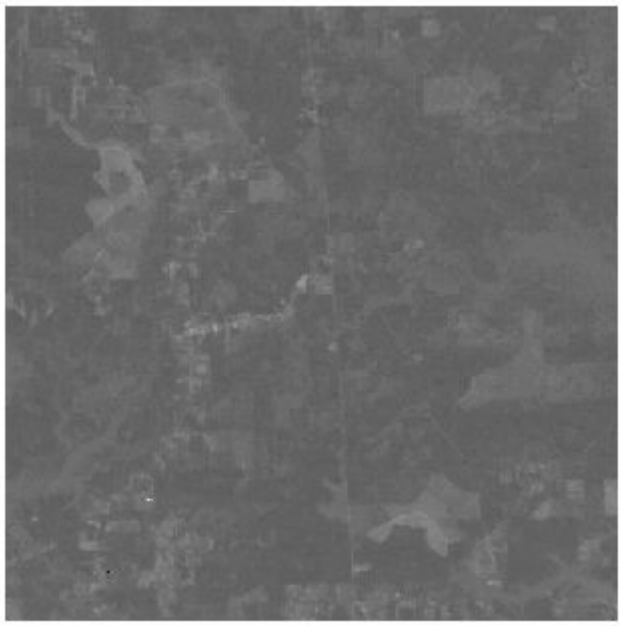


(a)

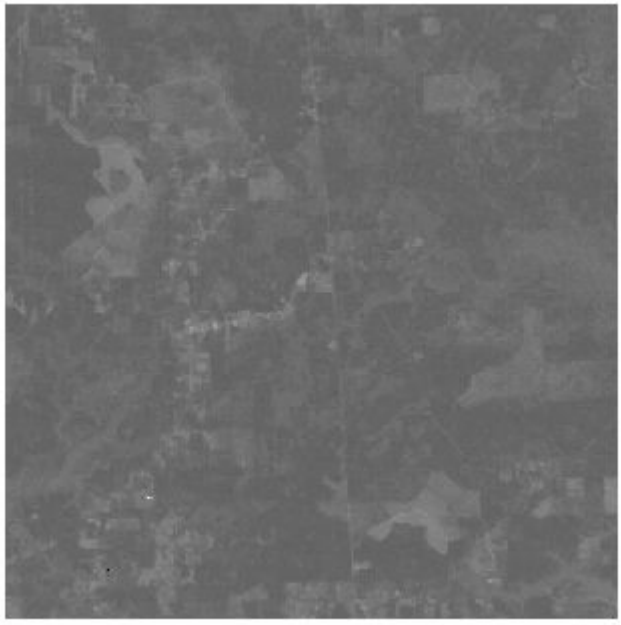


(b)

Figure A.32 Original and Restored images of Area-7, Band-4

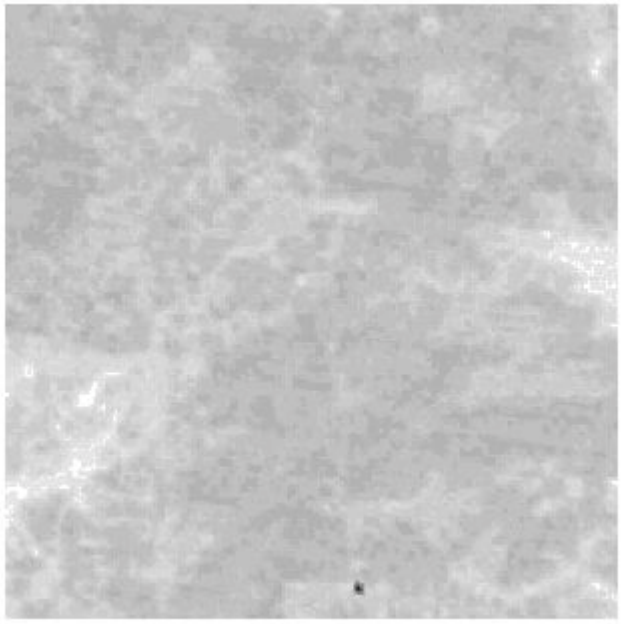


(a)

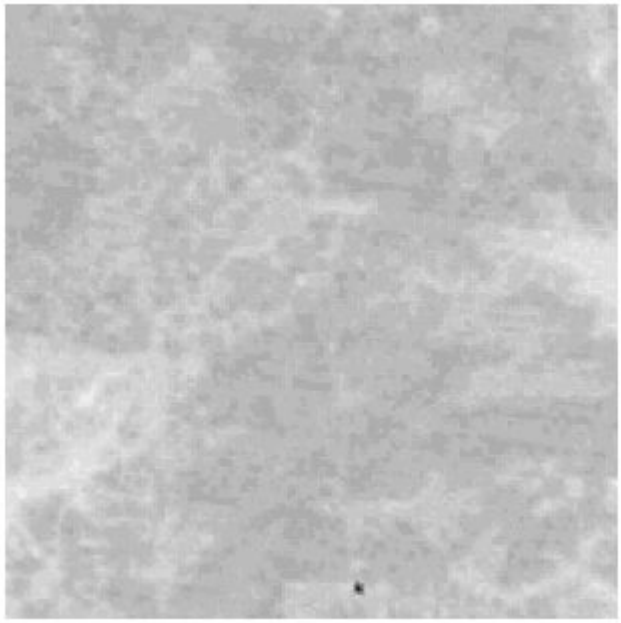


(b)

Figure A.33 Original and Restored images of Area-7, Band-5

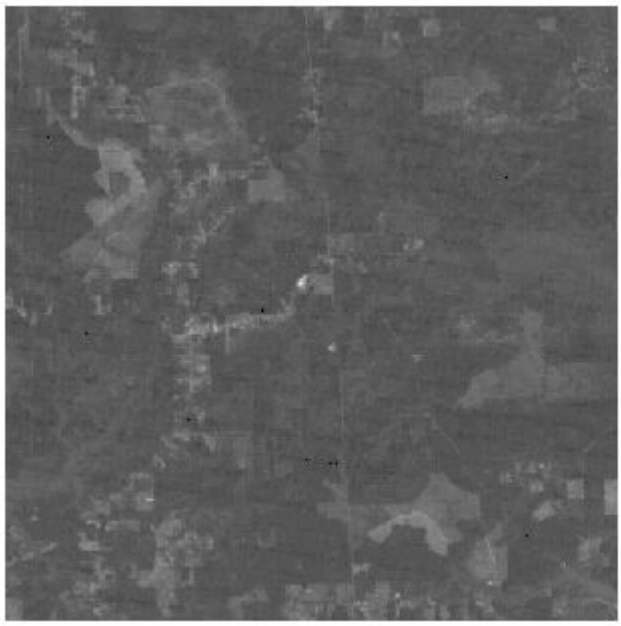


(a)

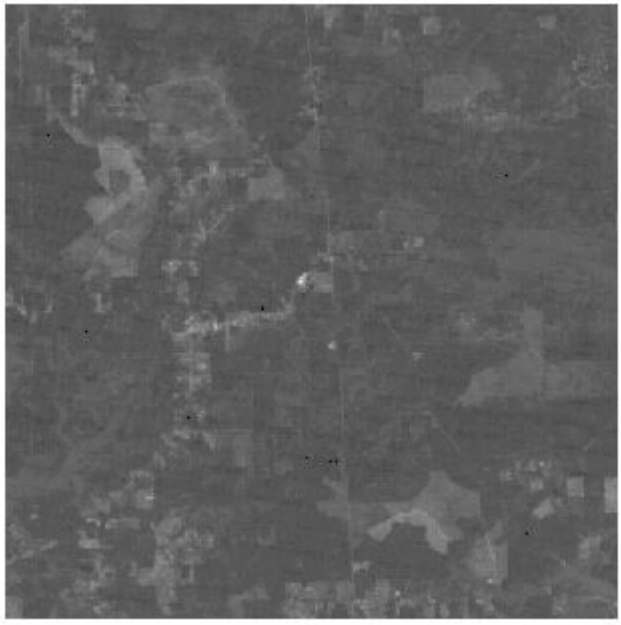


(b)

Figure A.34 Original and Restored images of Area-7, Band-6

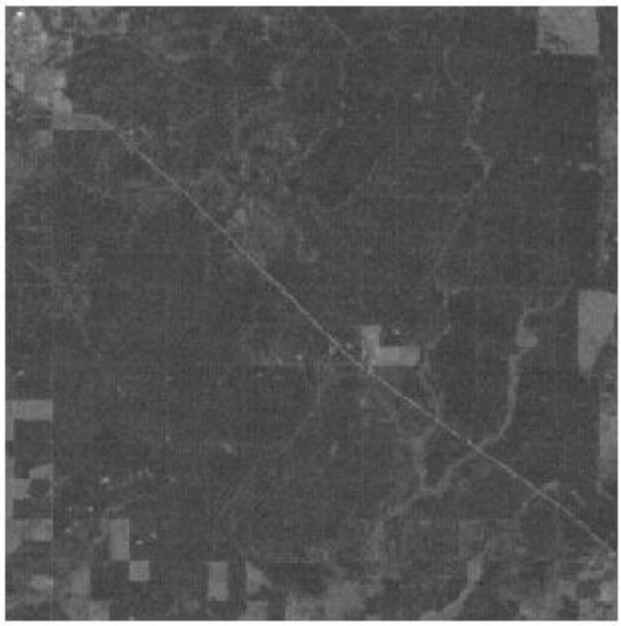


(a)

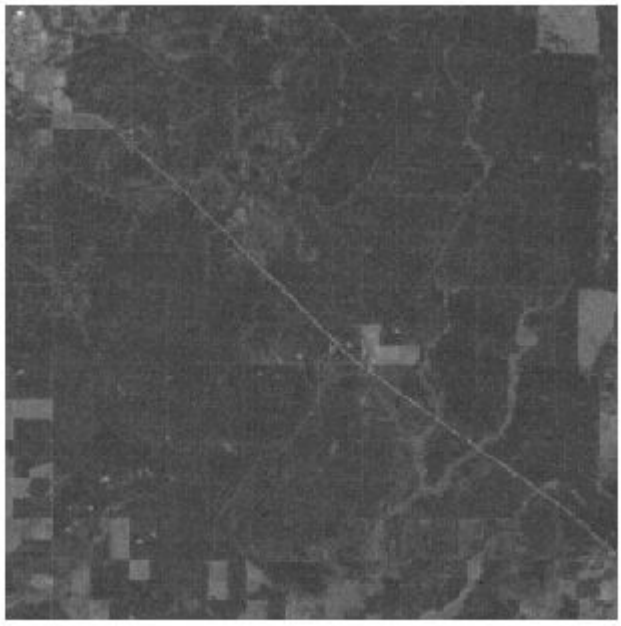


(b)

Figure A.35 Original and Restored images of Area-7, Band-7

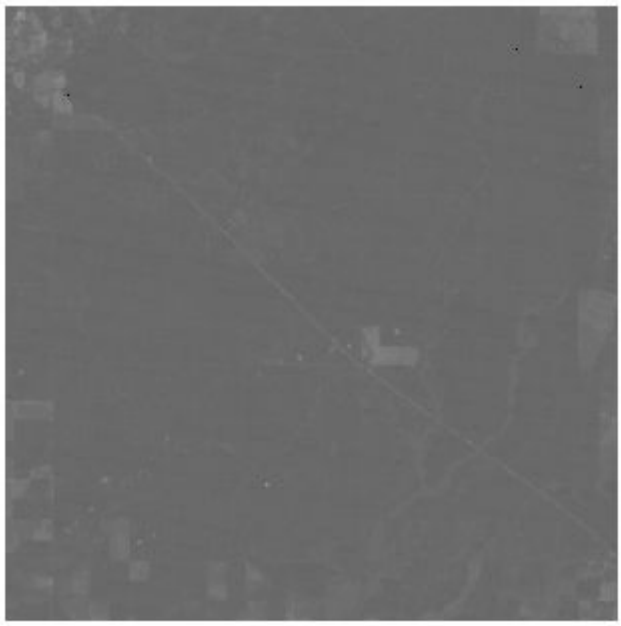


(a)

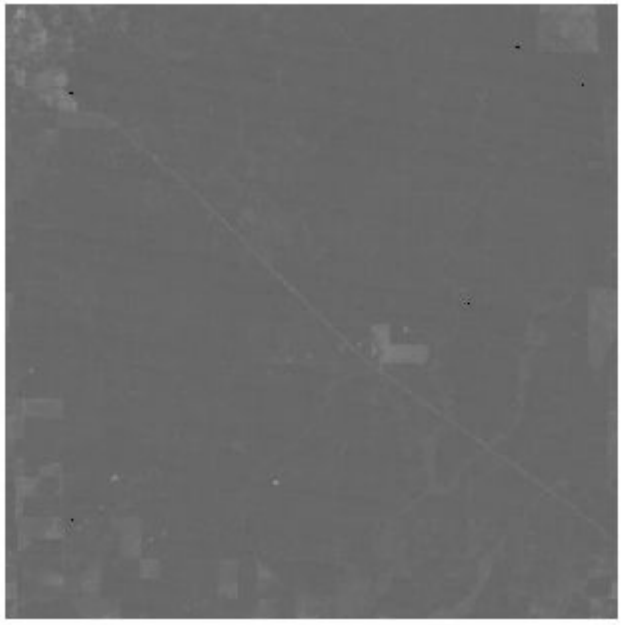


(b)

Figure A.36 Original and Restored images of Area-8, Band-1

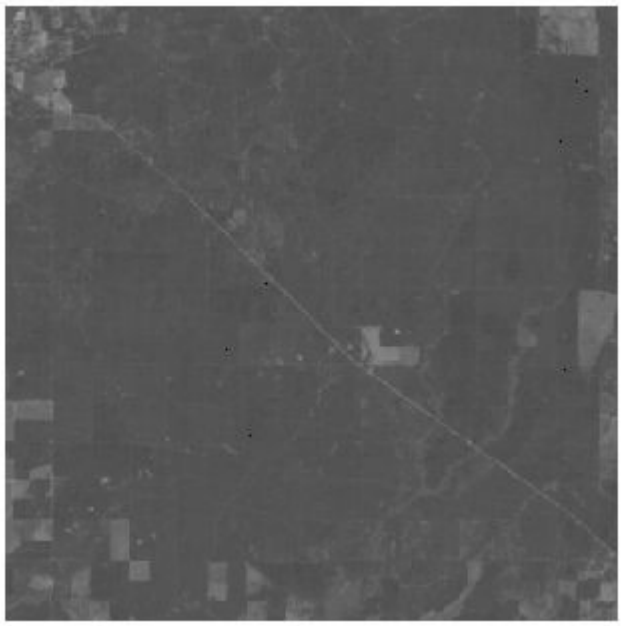


(b)

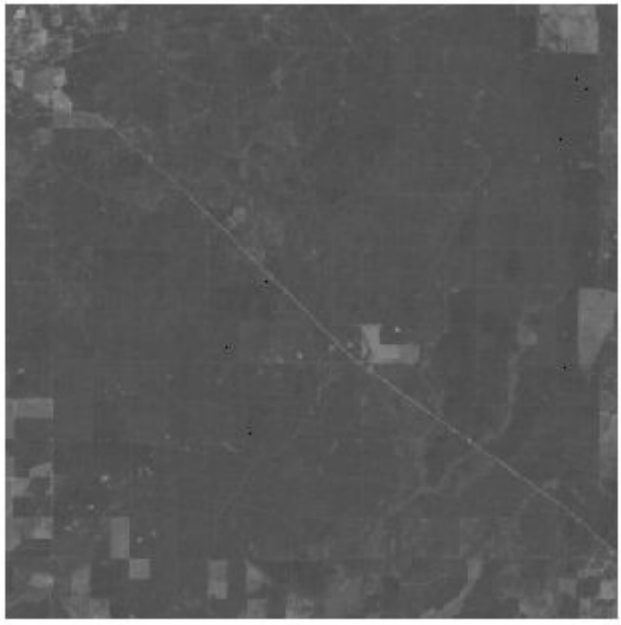


(b)

Figure A.37 Original and Restored images of Area-8, Band-2

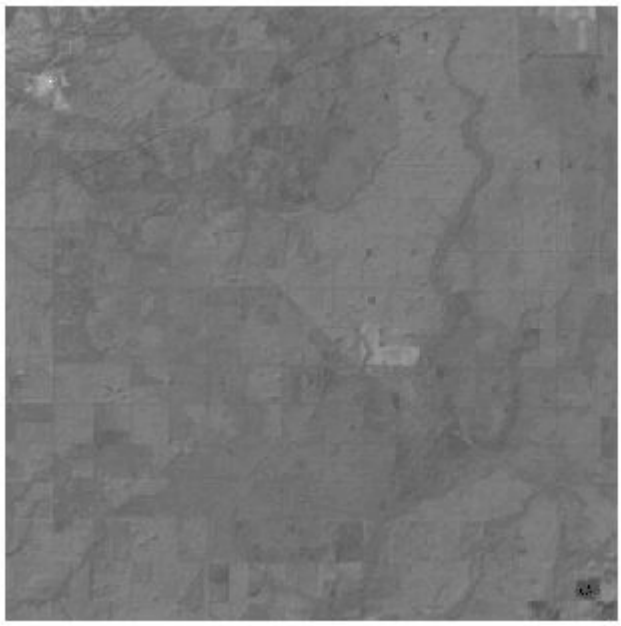


(a)

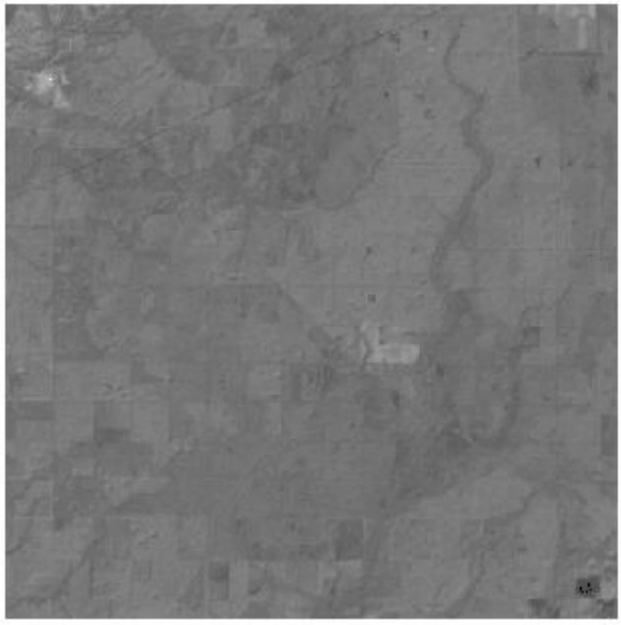


(b)

Figure A.38 Original and Restored images of Area-8, Band-3

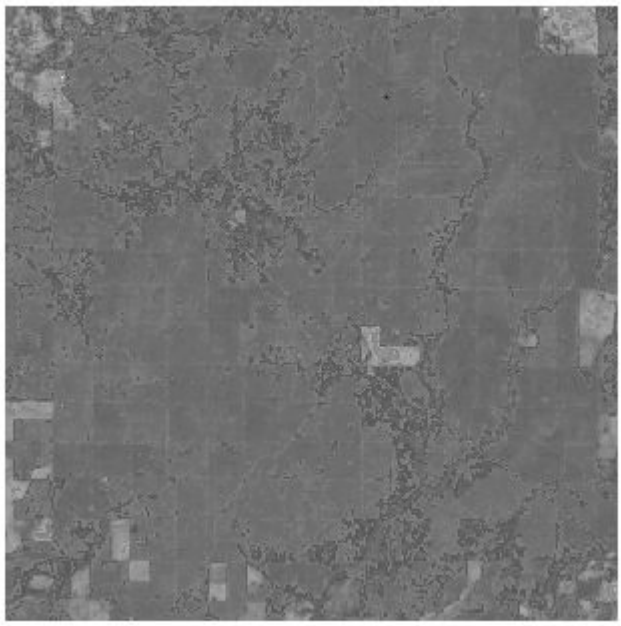


(a)

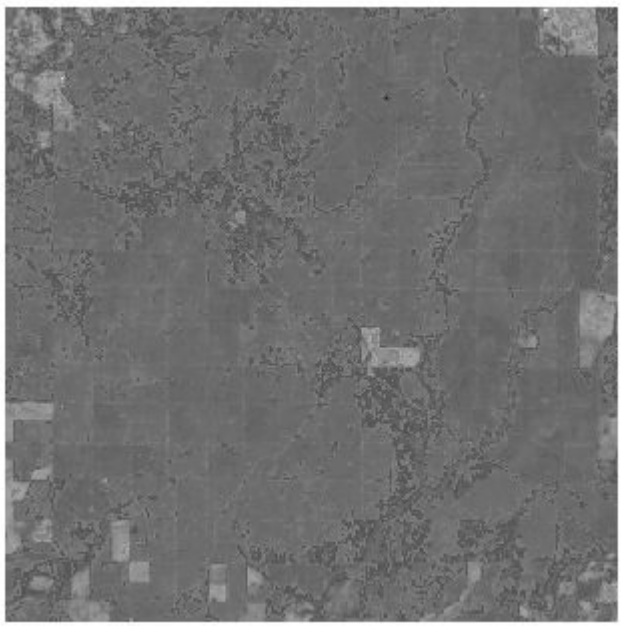


(b)

Figure A.39 Original and Restored images of Area-8, Band-4

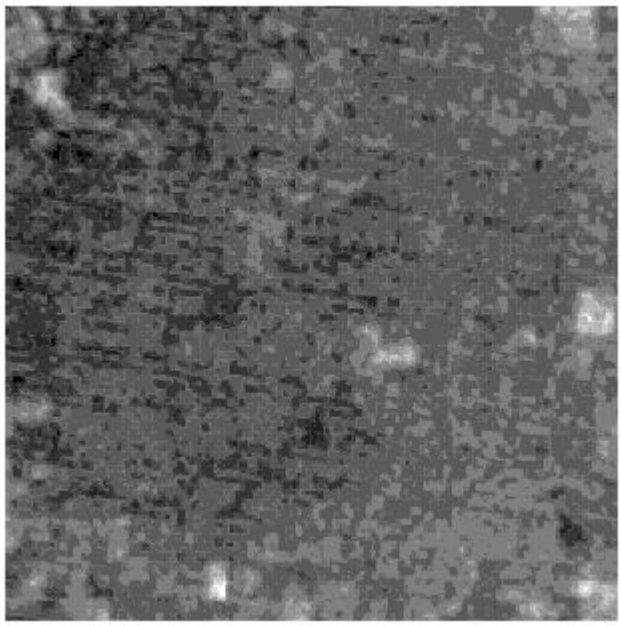


(a)

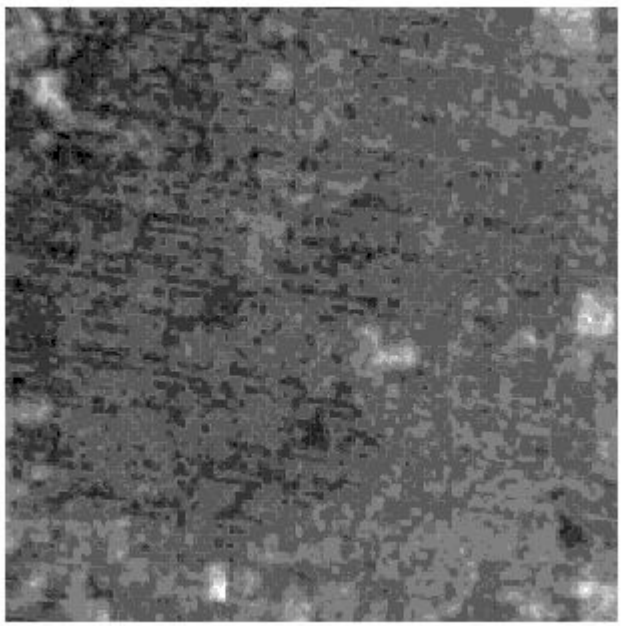


(b)

Figure A.40 Original and Restored images of Area-8, Band-5



(b)



(b)

Figure A.41 Original and Restored images of Area-8, Band-6

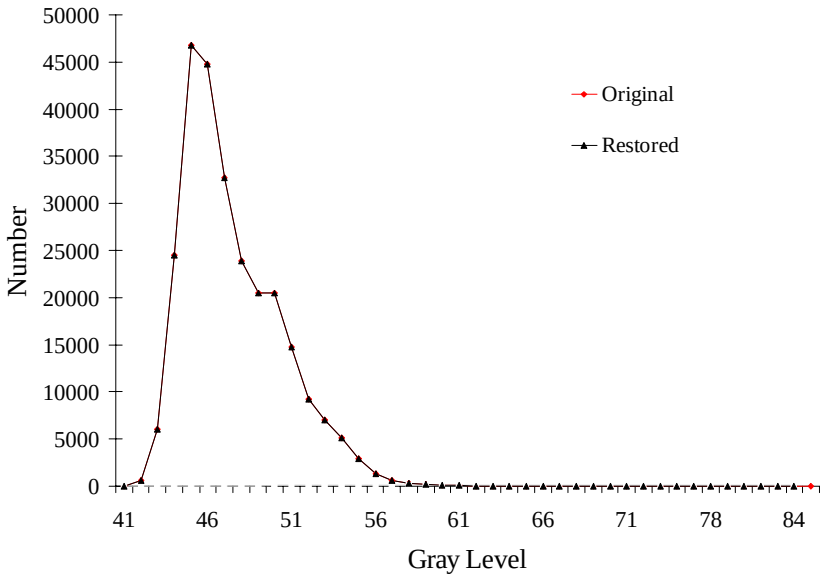
(a)

(b)

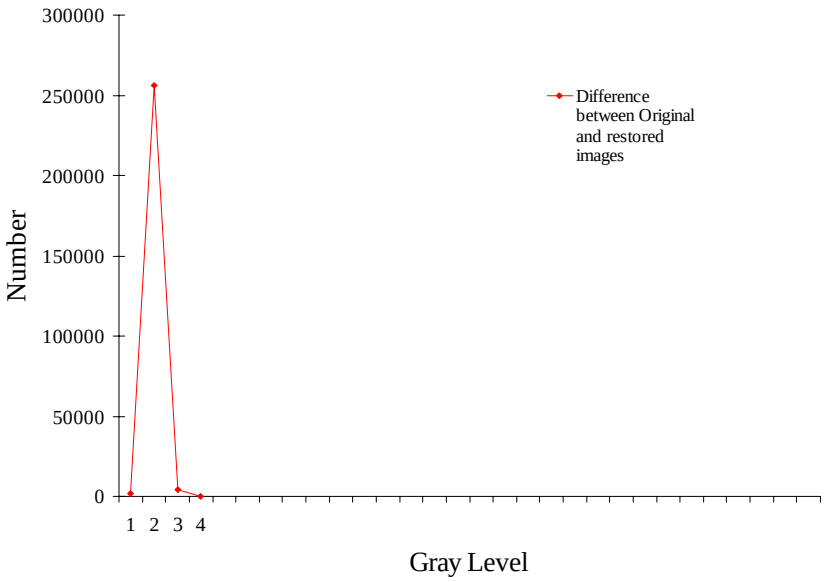
Figure A.42 Original and Restored images of Area-8, Band-7

APPENDIX B

PROFILES AND DIFFERENCE IMAGES

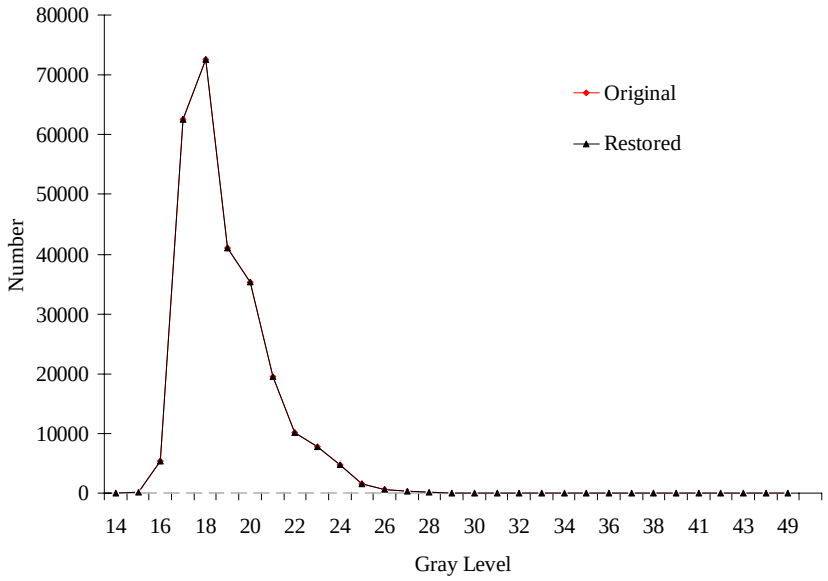


(a)

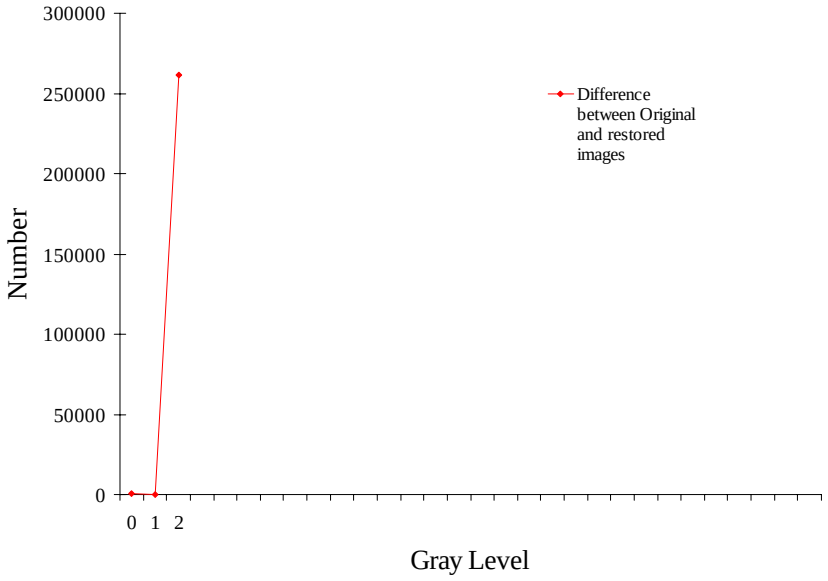


(b)

Figure B.1 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 1

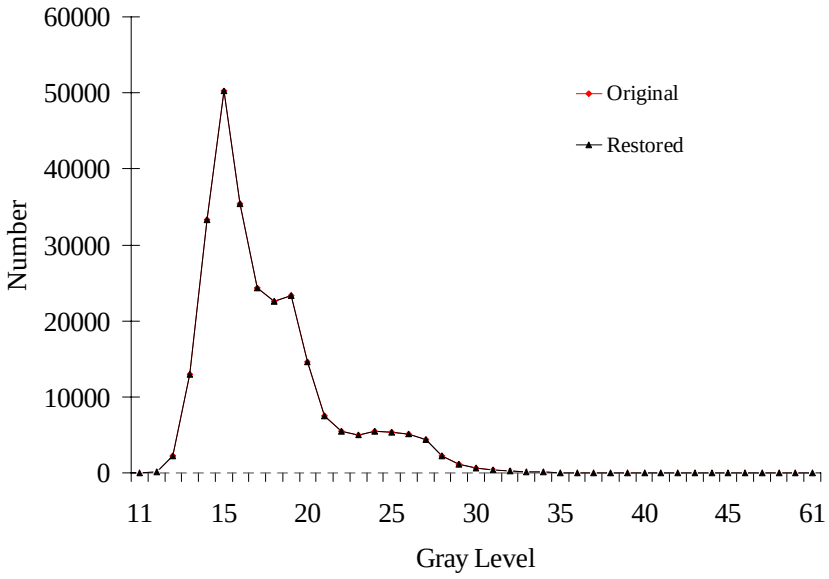


(a)

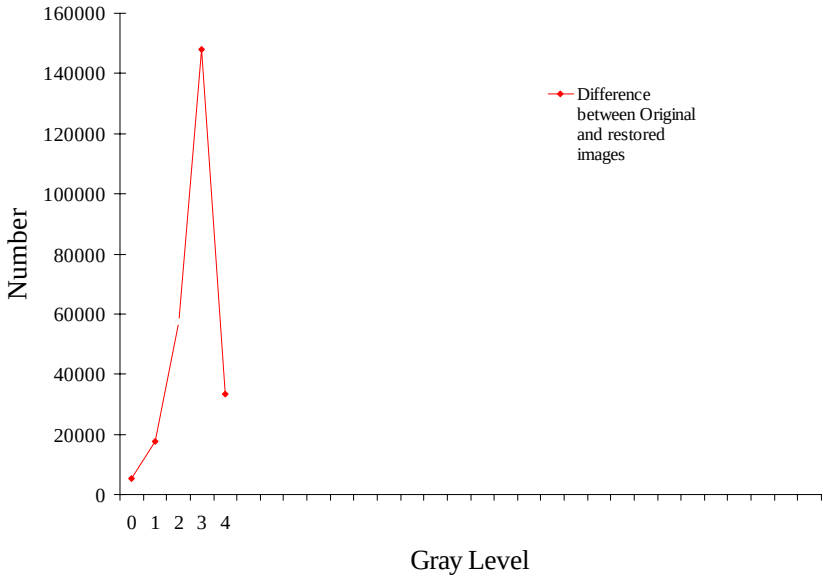


(b)

Figure B.2 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 2

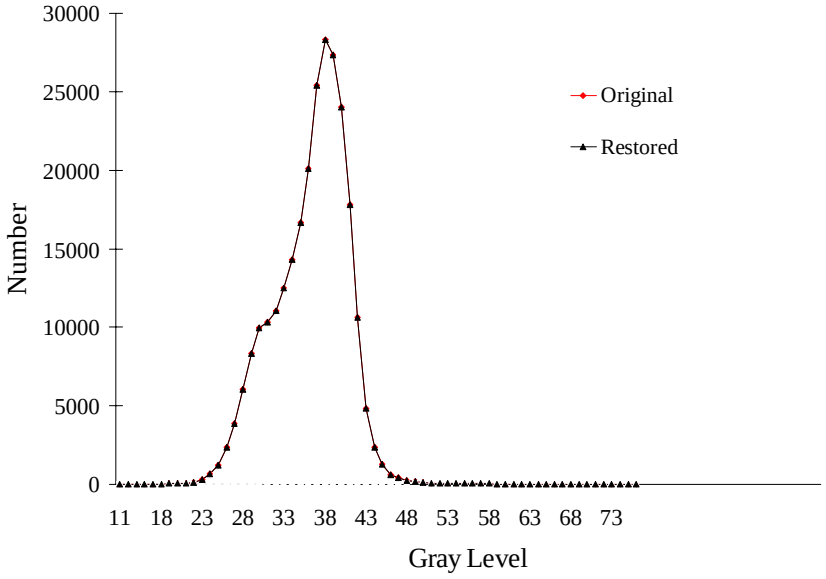


(a)

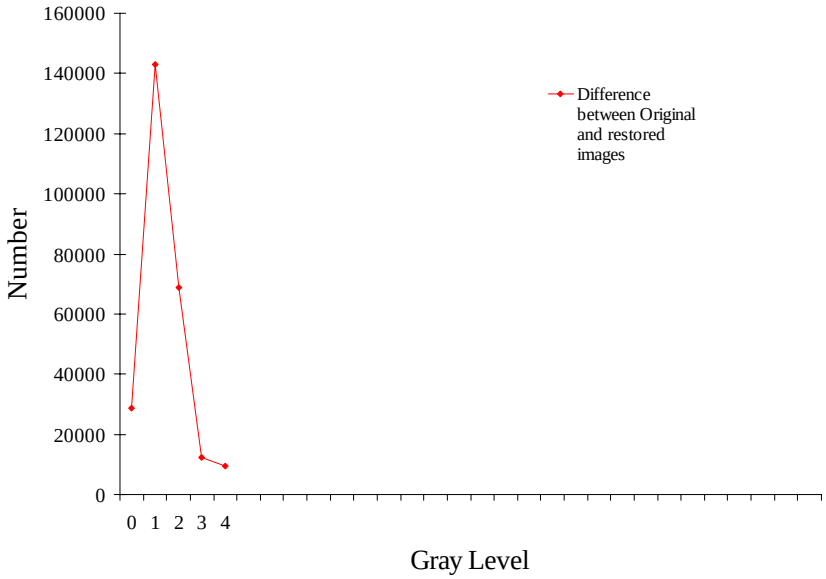


(b)

Figure B.3 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 3

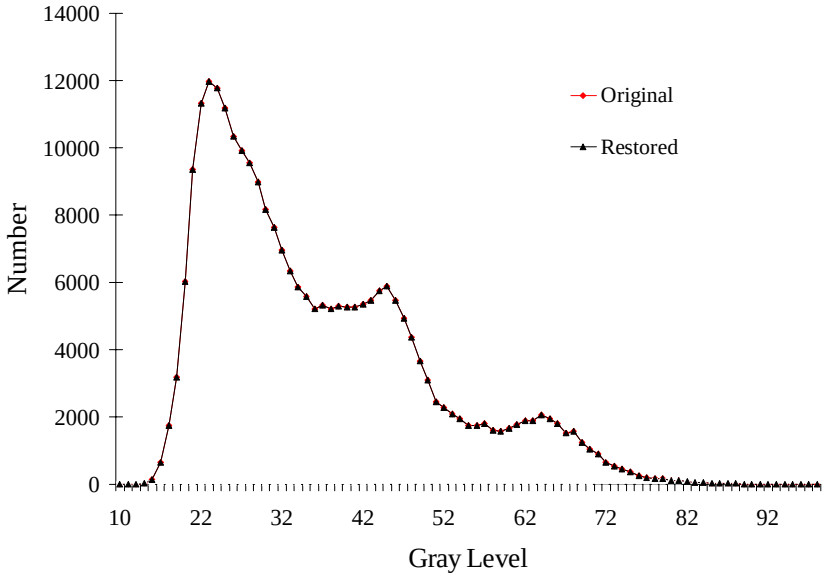


(a)

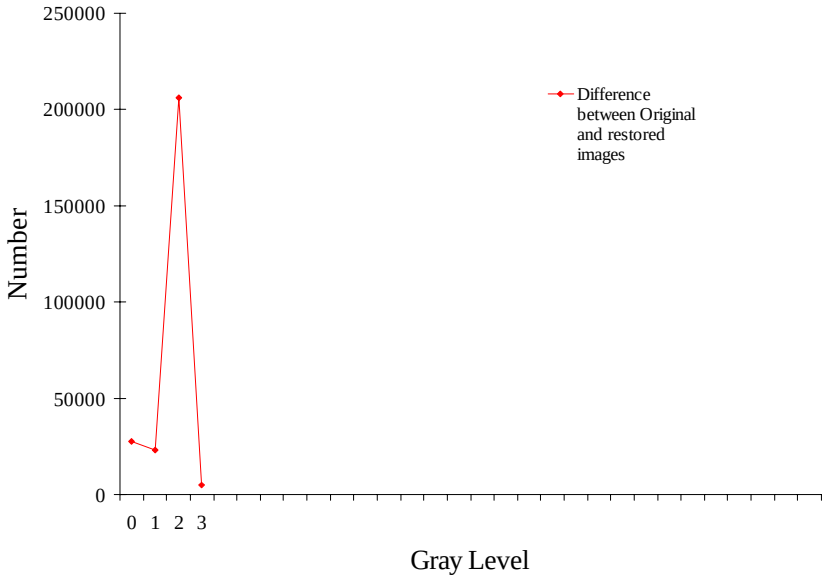


(b)

Figure B.4 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 4

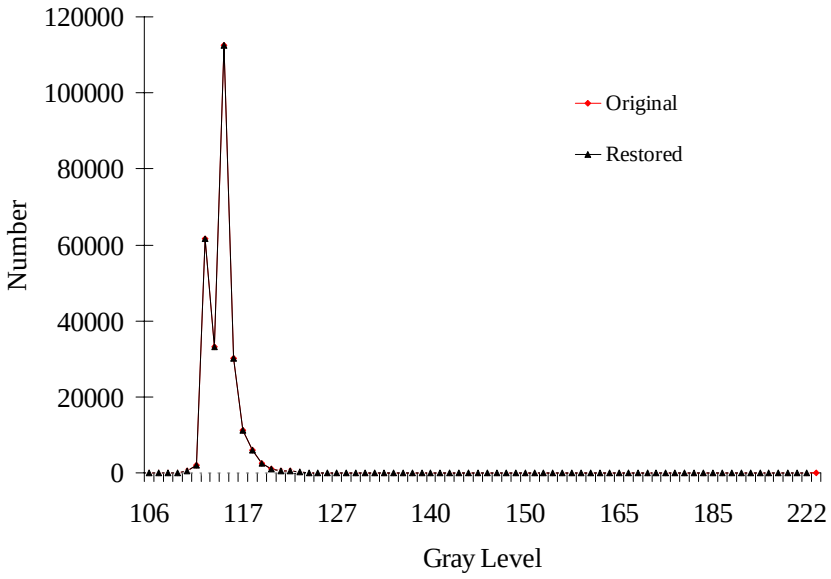


(a)

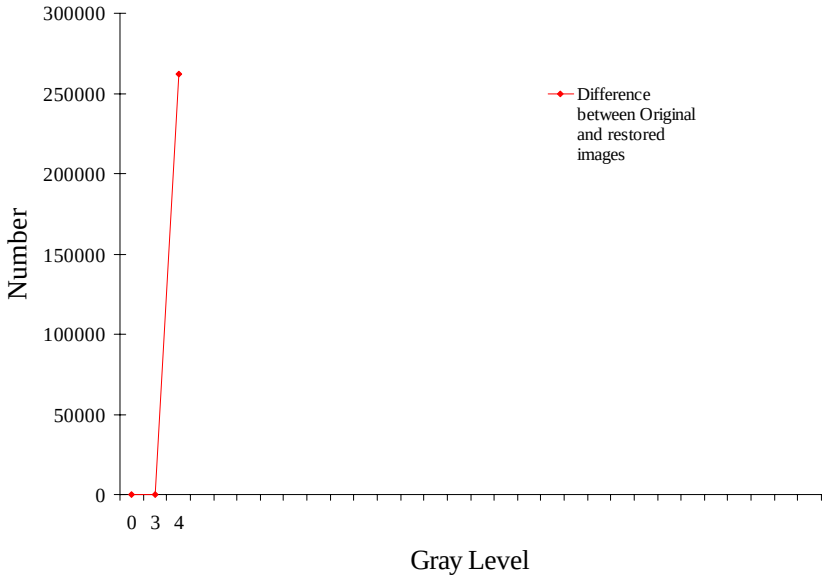


(b)

Figure B.5 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 5

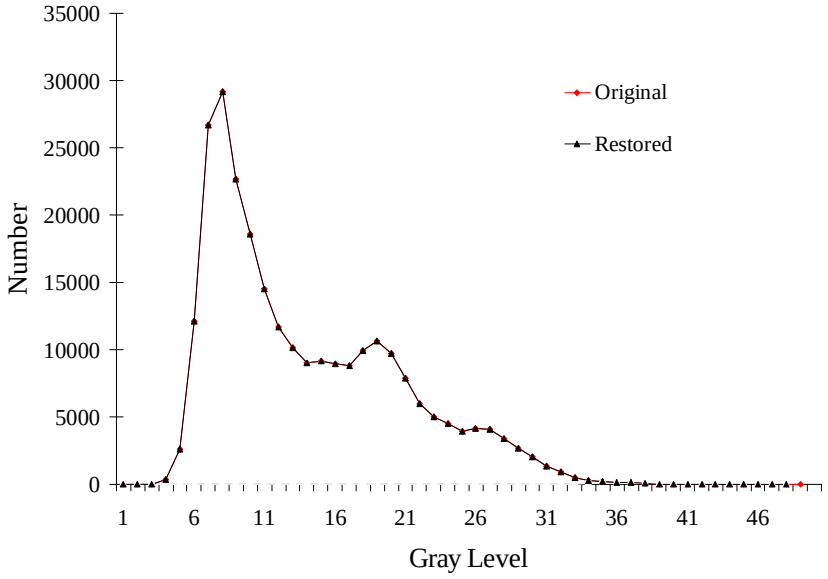


(a)

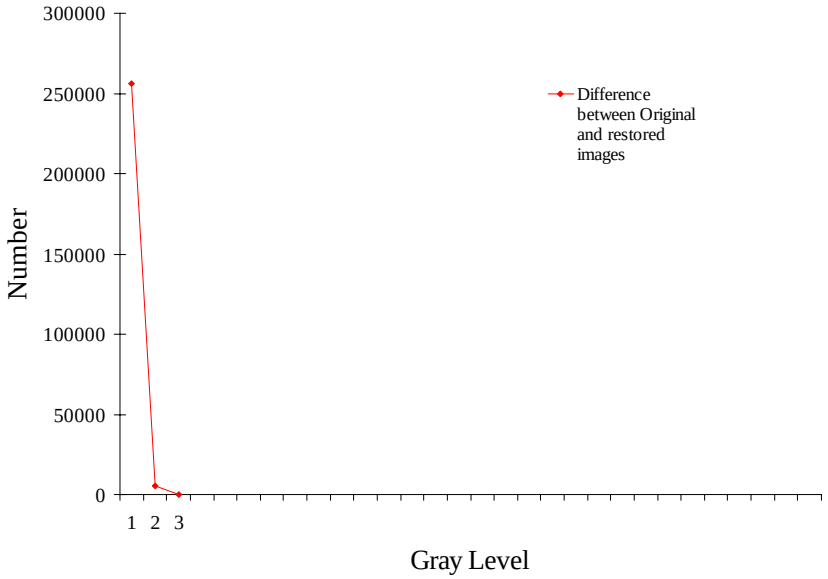


(b)

Figure B.6 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 6

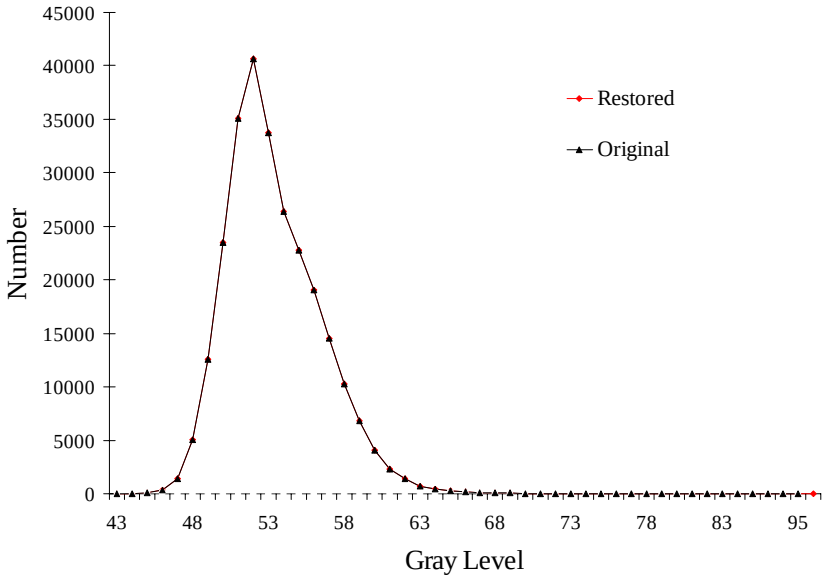


(a)

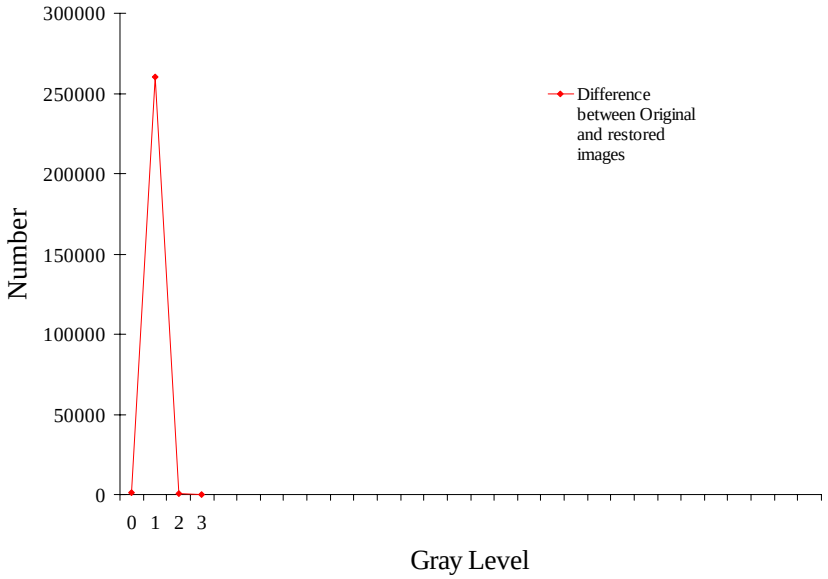


(b)

Figure B.7 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-1, Band 7

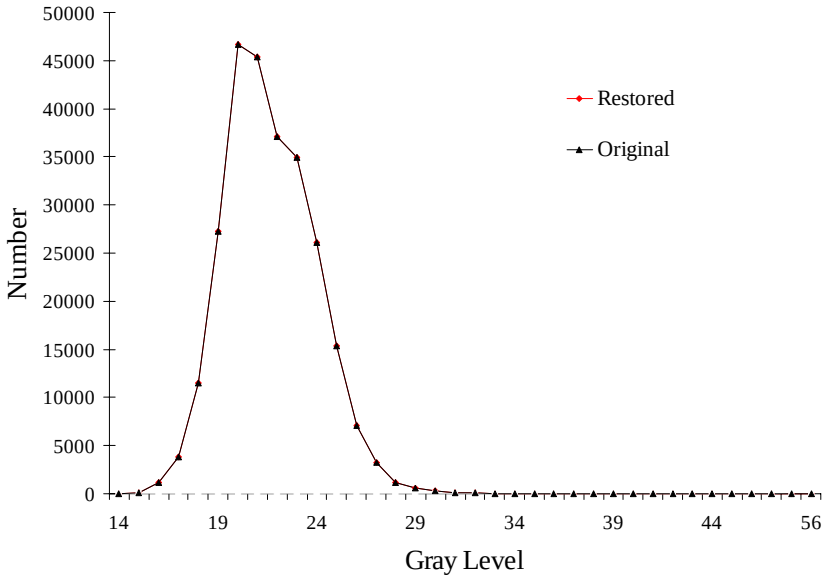


(a)

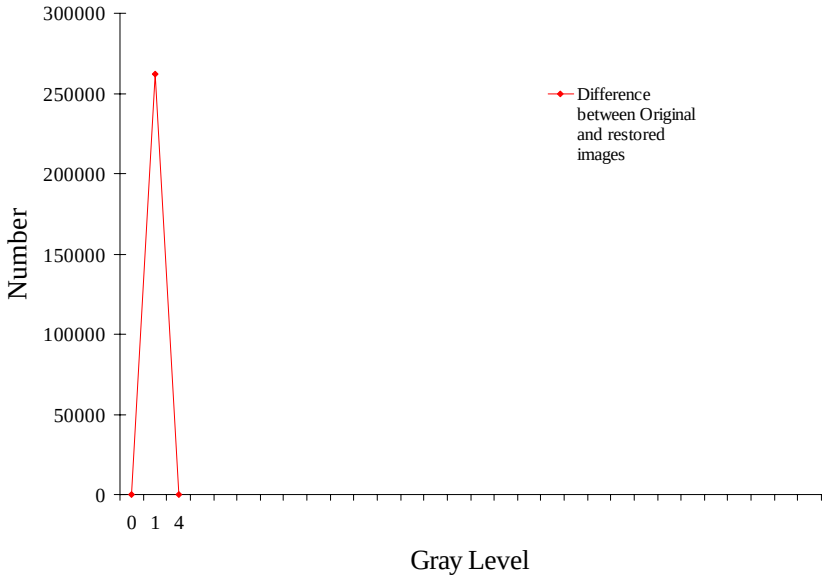


(b)

Figure B.8 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 1

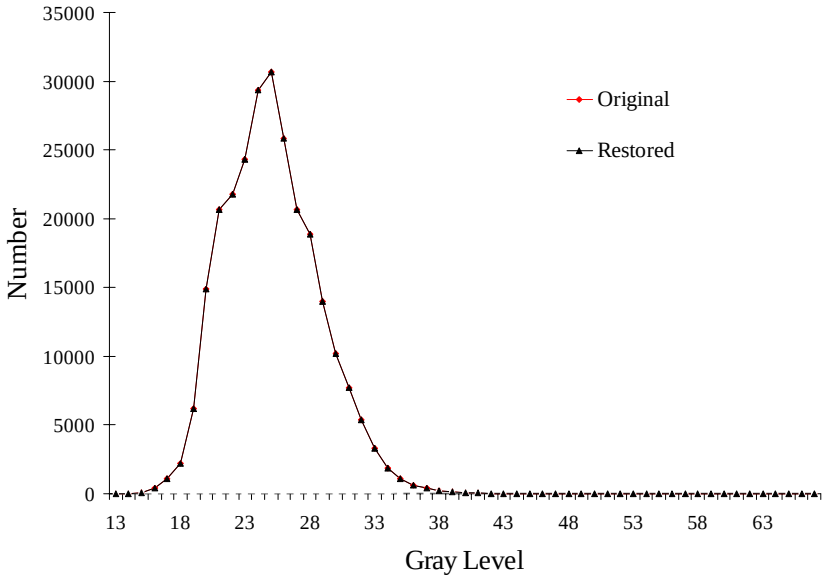


(a)

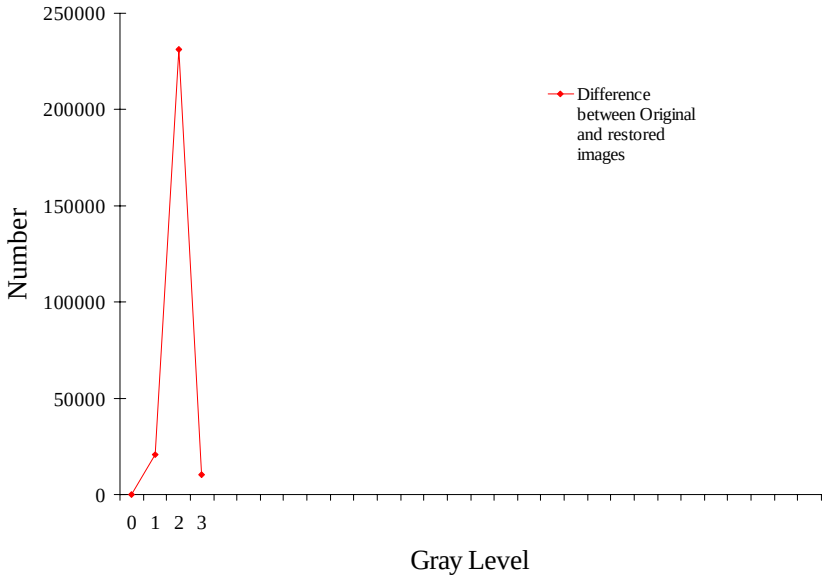


(b)

Figure B.9 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 2

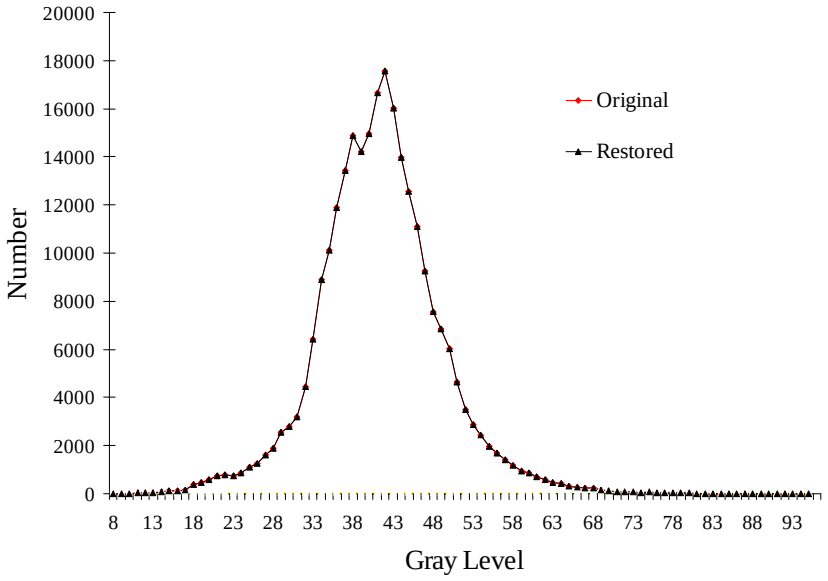


(a)

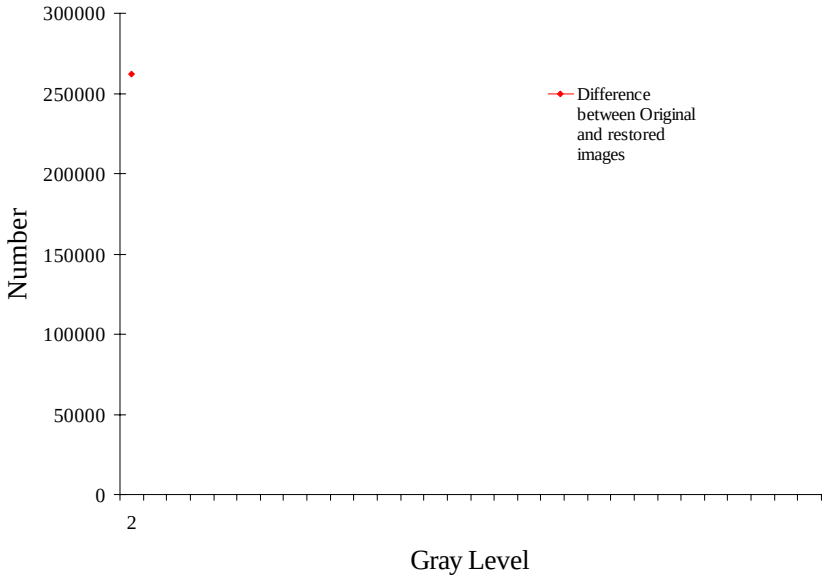


(b)

Figure B.10 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 3

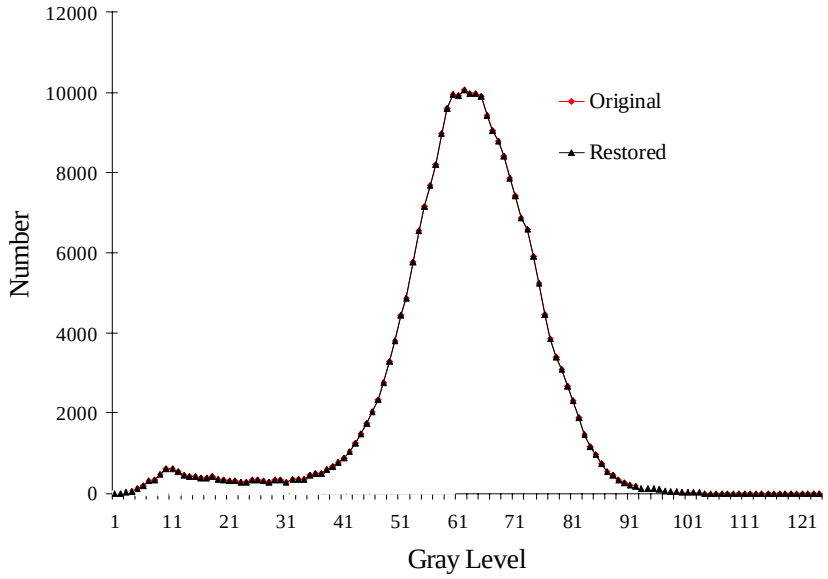


(a)

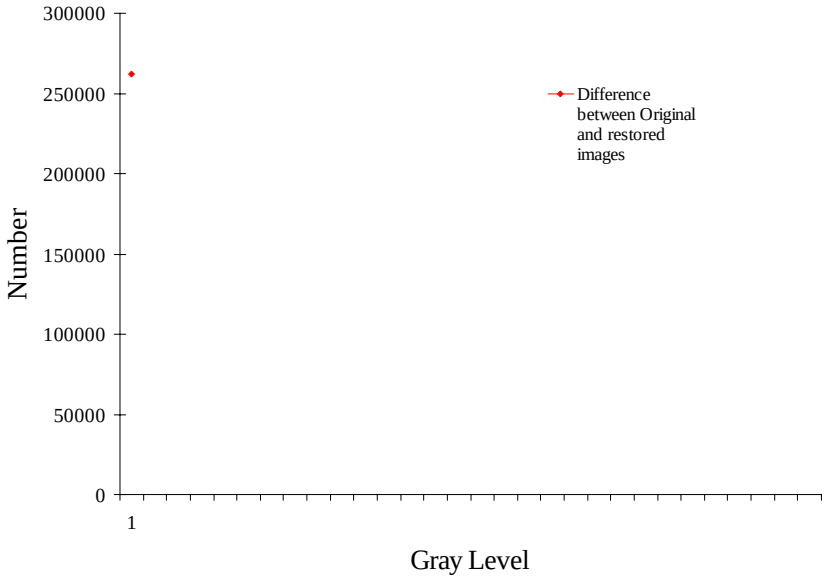


(b)

Figure B.11 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 4

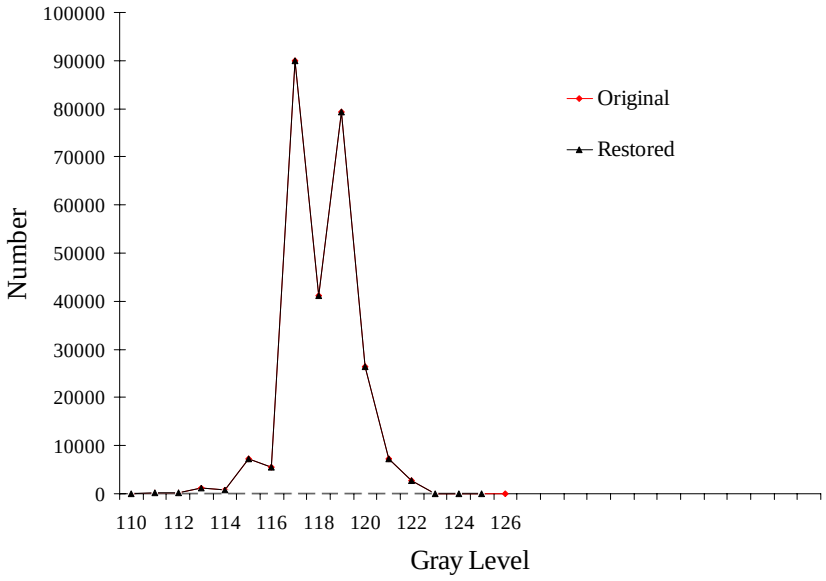


(a)

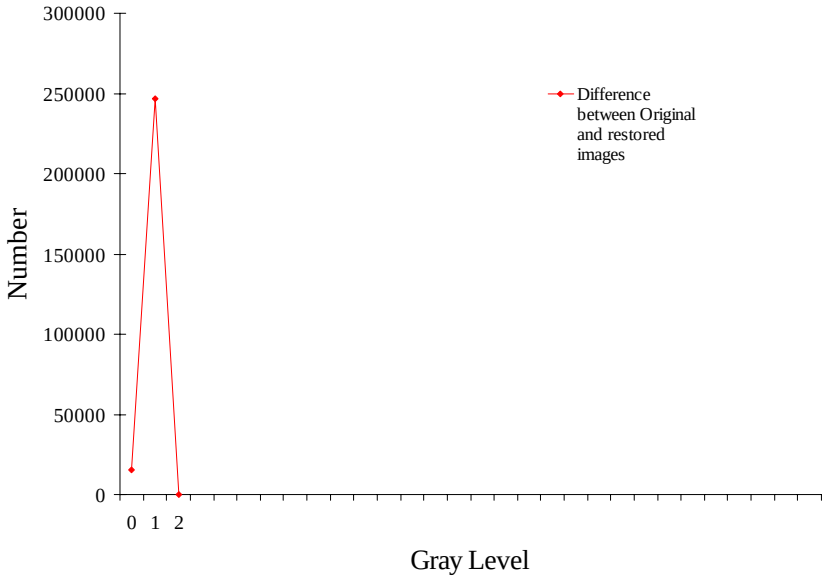


(b)

Figure B.12 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 5

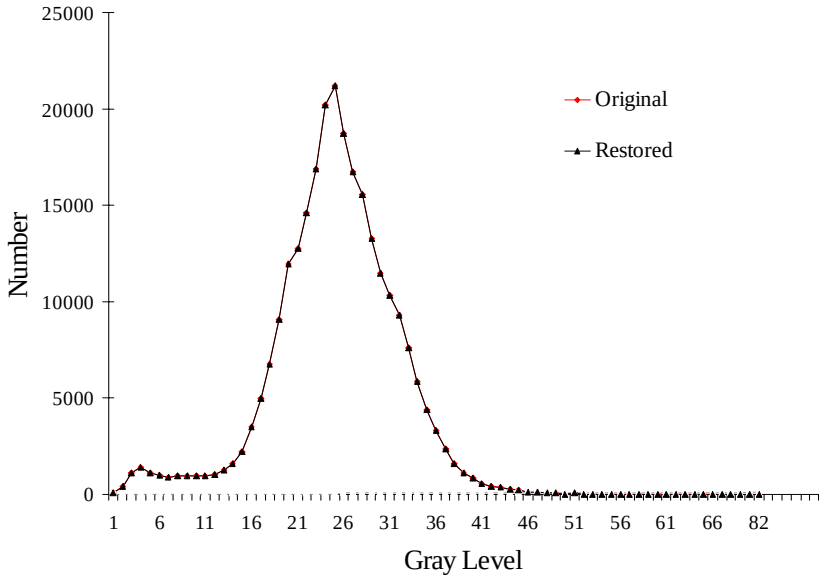


(a)

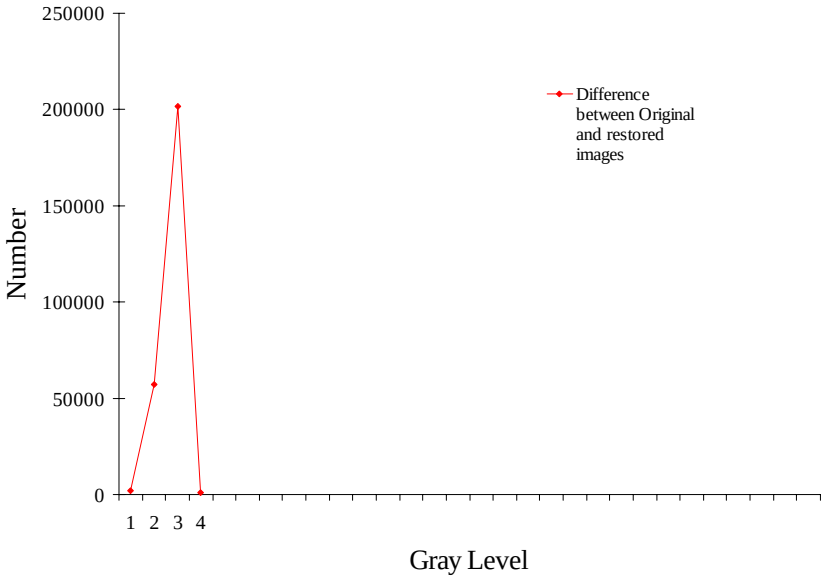


(b)

Figure B.13 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 6

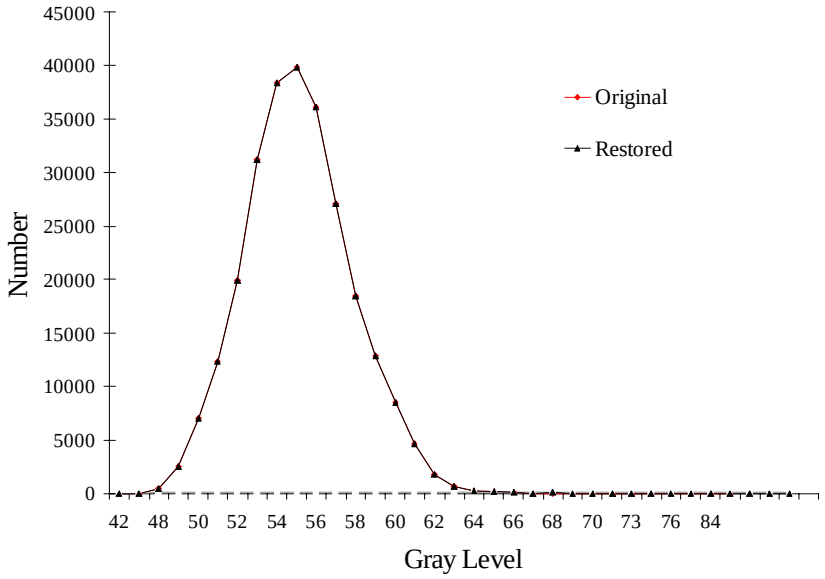


(a)

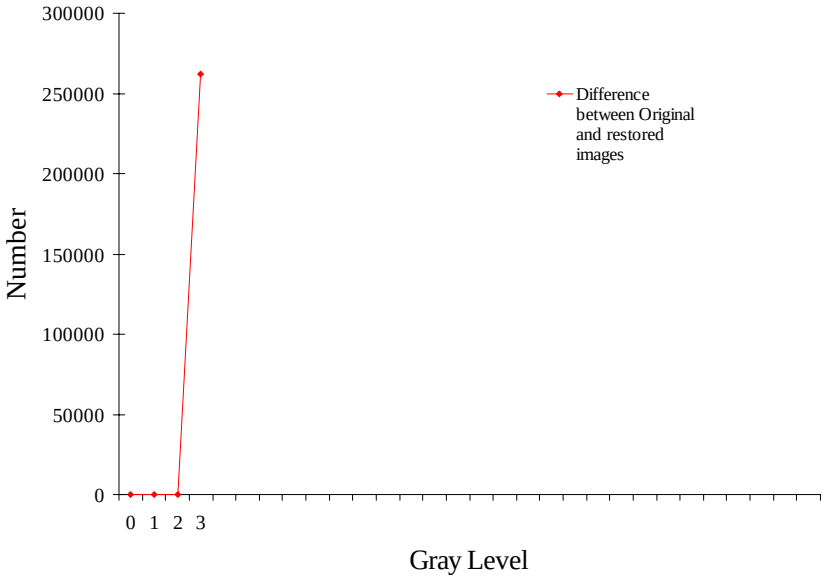


(b)

Figure B.14 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-2, Band 7

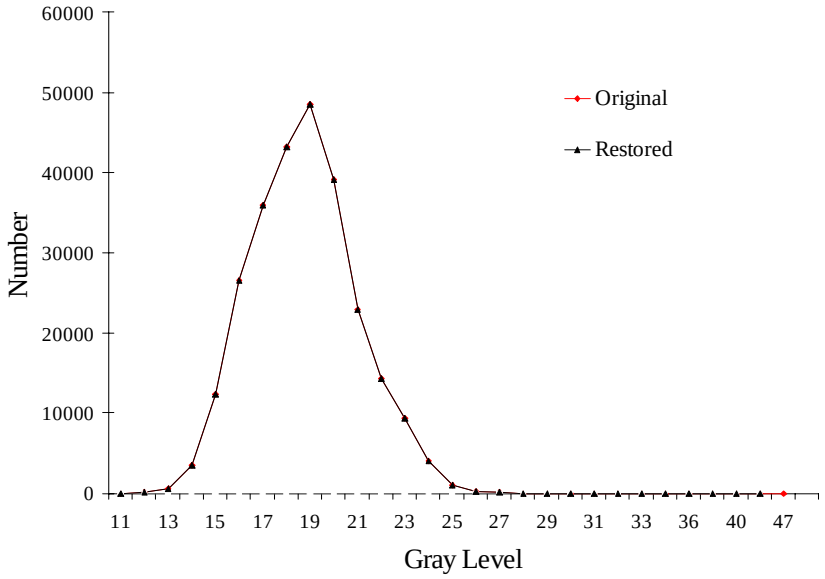


(a)

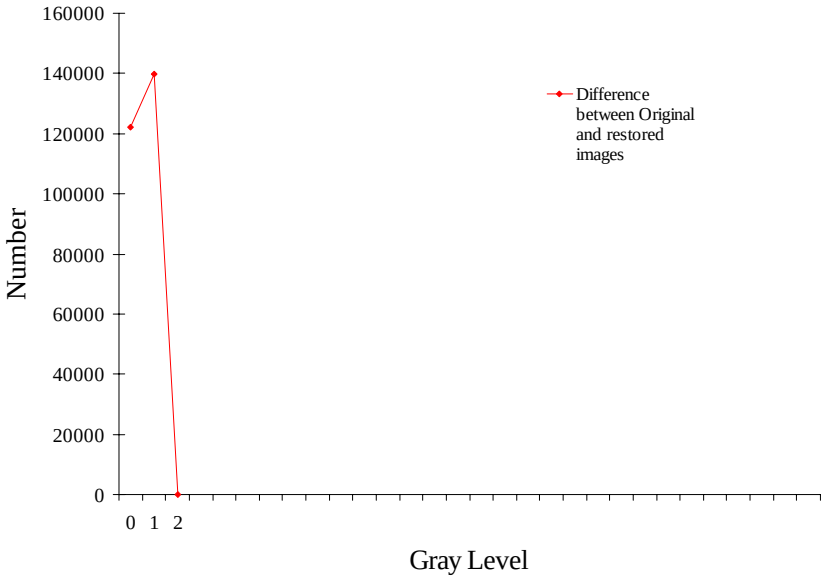


(b)

Figure B.15 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 1

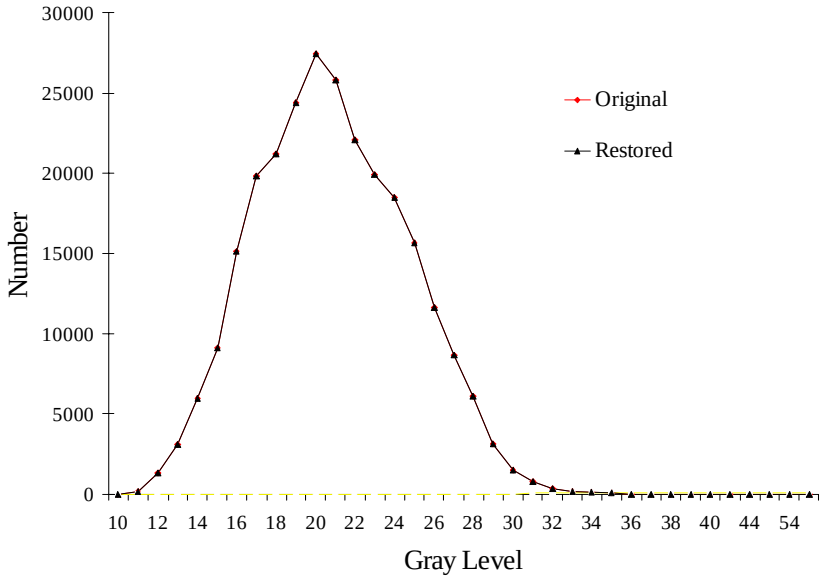


(a)

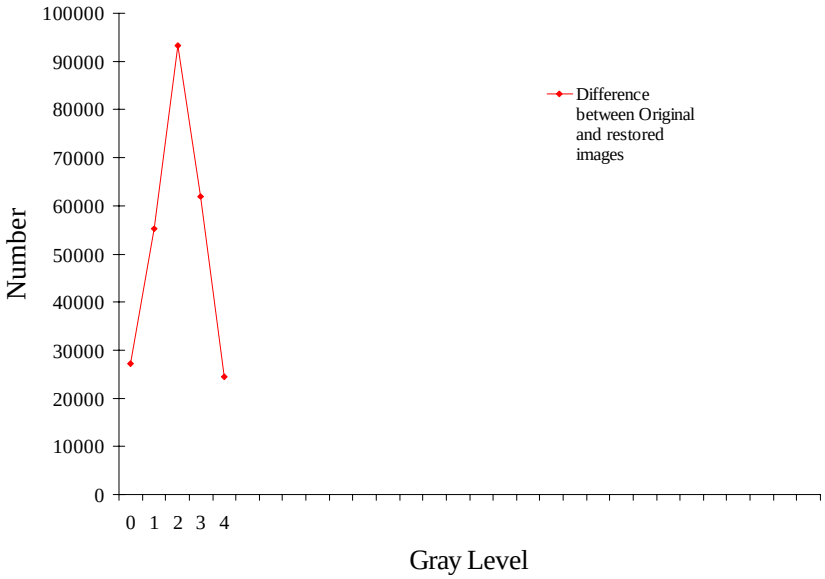


(b)

Figure B.16 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 2

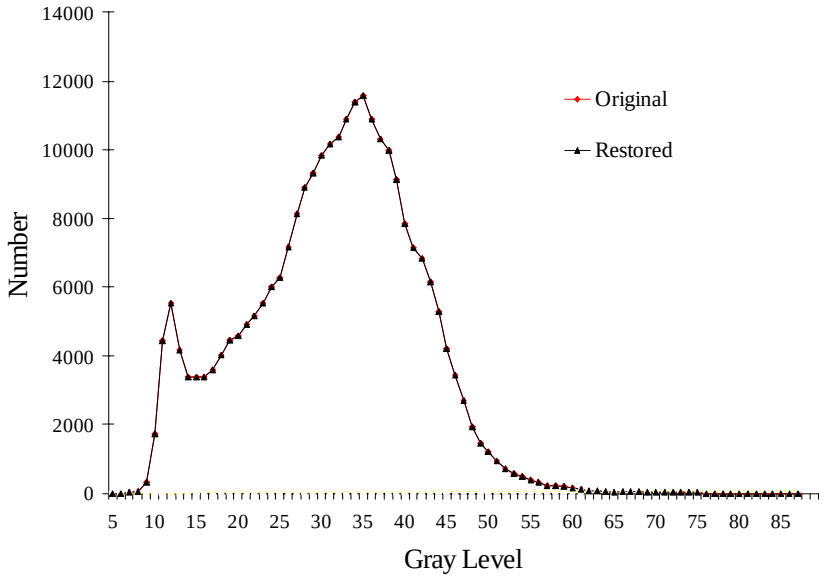


(a)

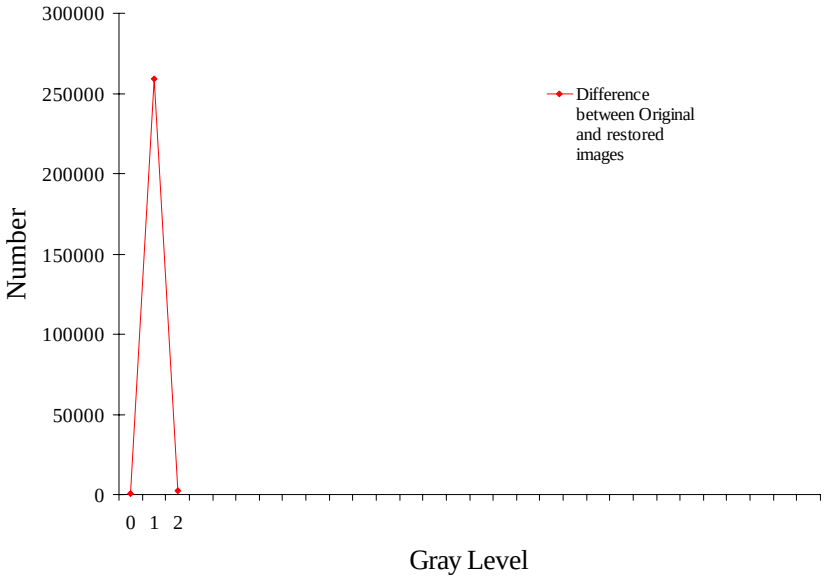


(b)

Figure B.17 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 3

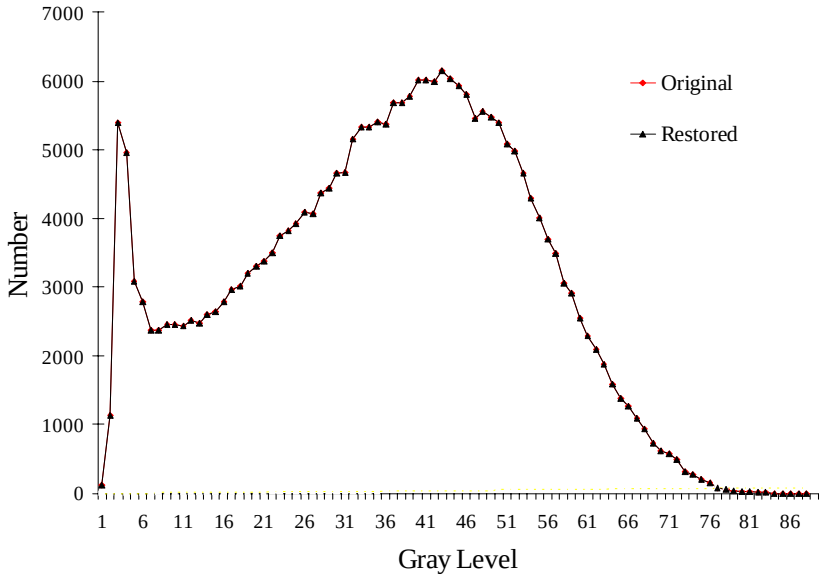


(a)

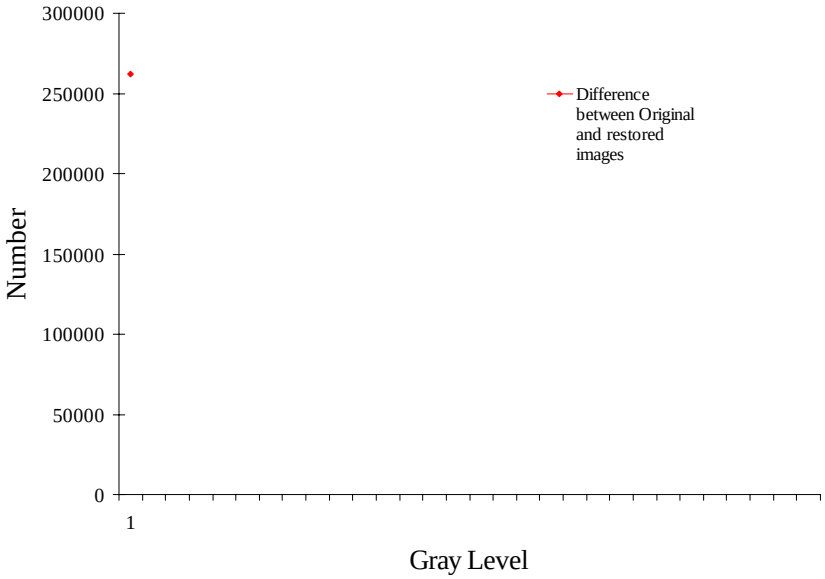


(b)

Figure B.18 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 4

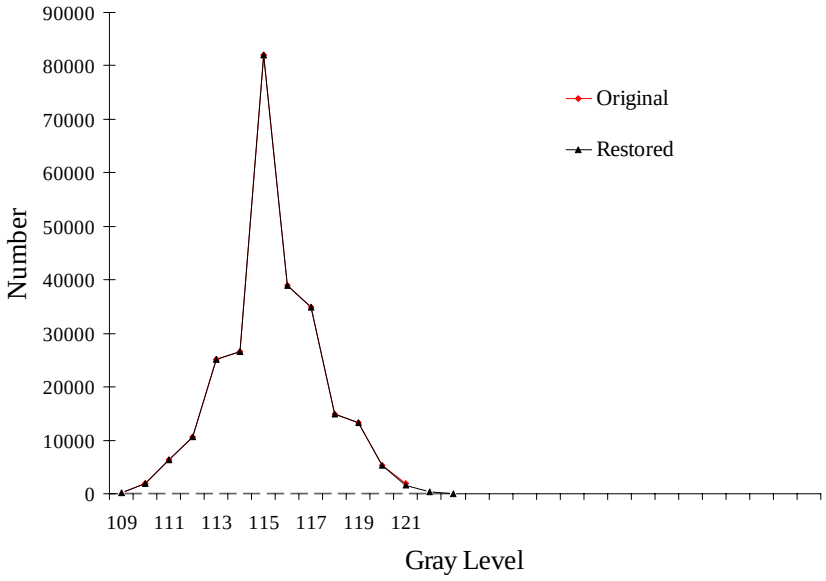


(a)

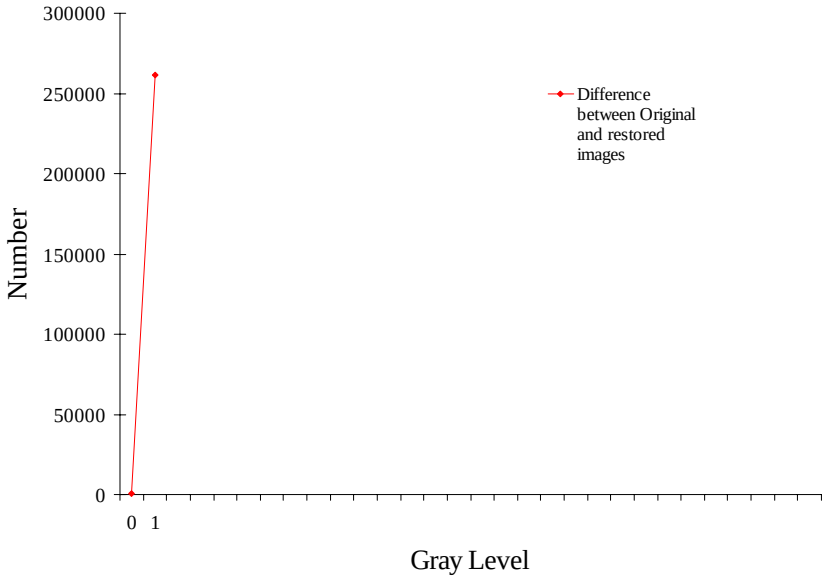


(b)

Figure B.19 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 5

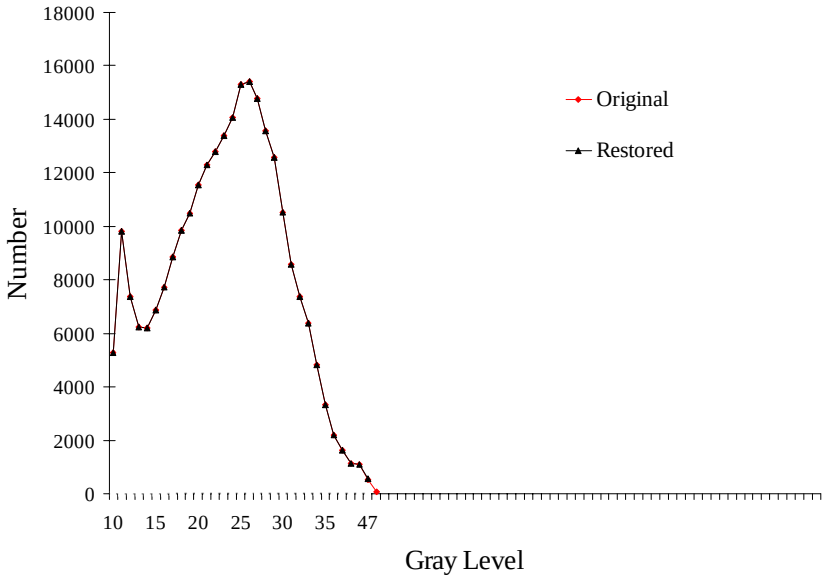


(a)

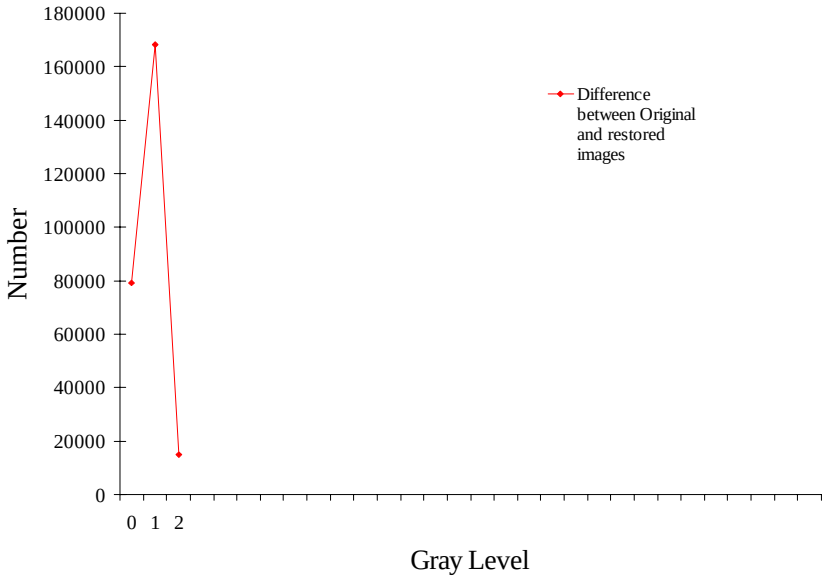


(b)

Figure B.20 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 6

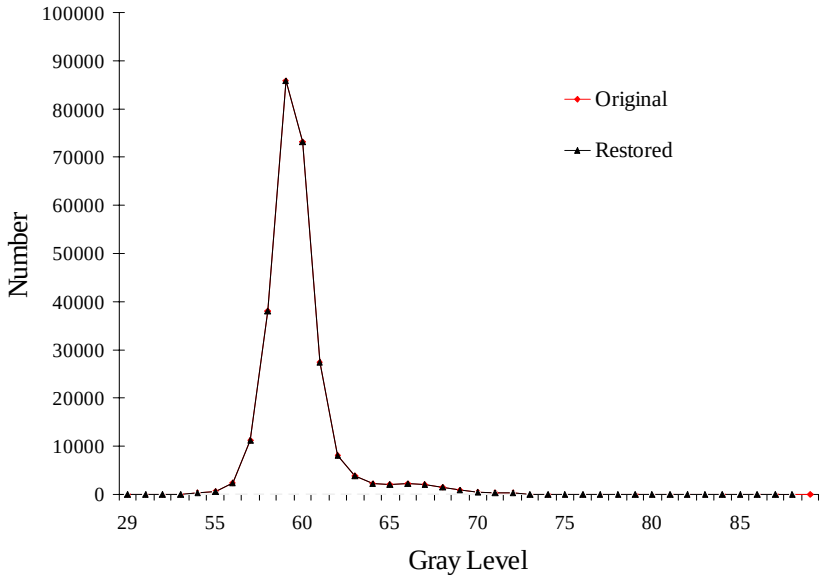


(a)

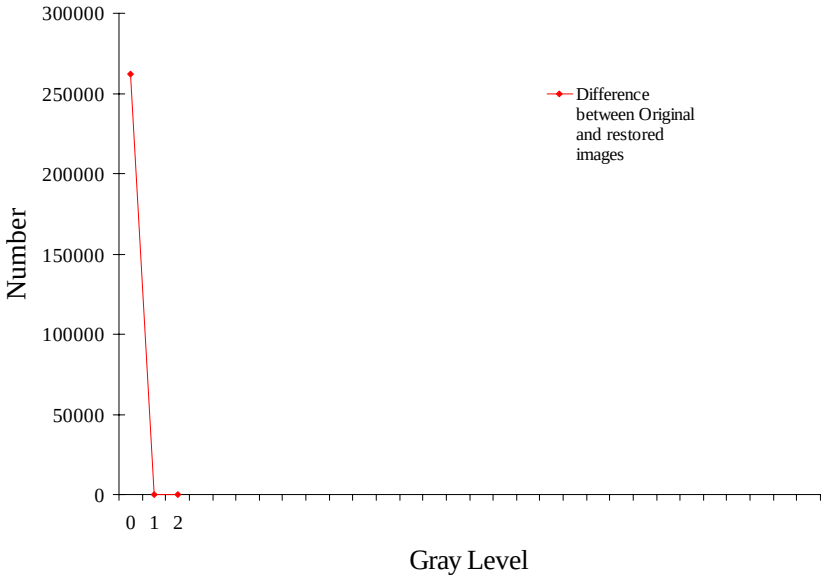


(b)

Figure B.21 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-3, Band 7

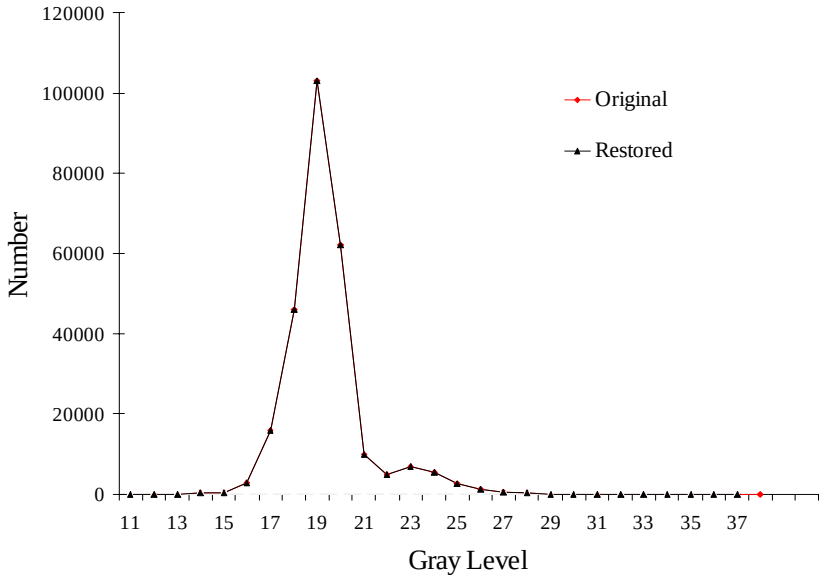


(a)

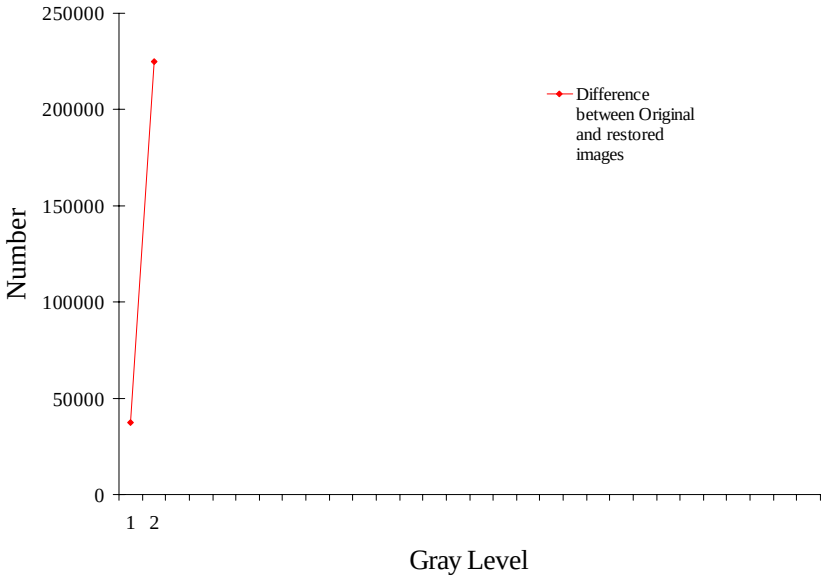


(b)

Figure B.22 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 1

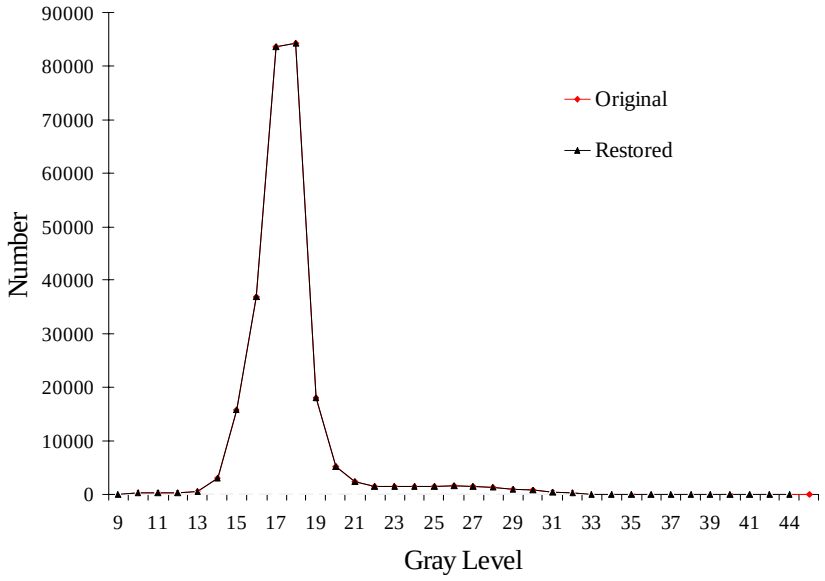


(a)

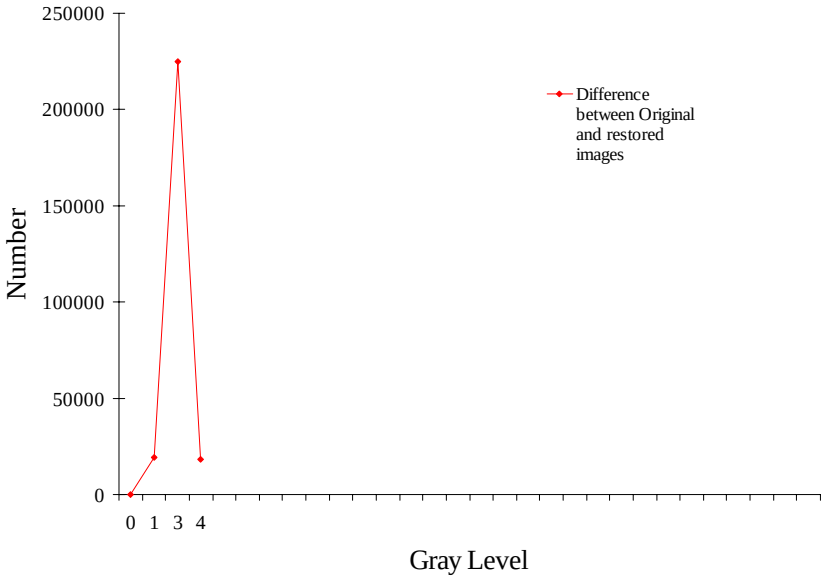


(b)

Figure B.23 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 2

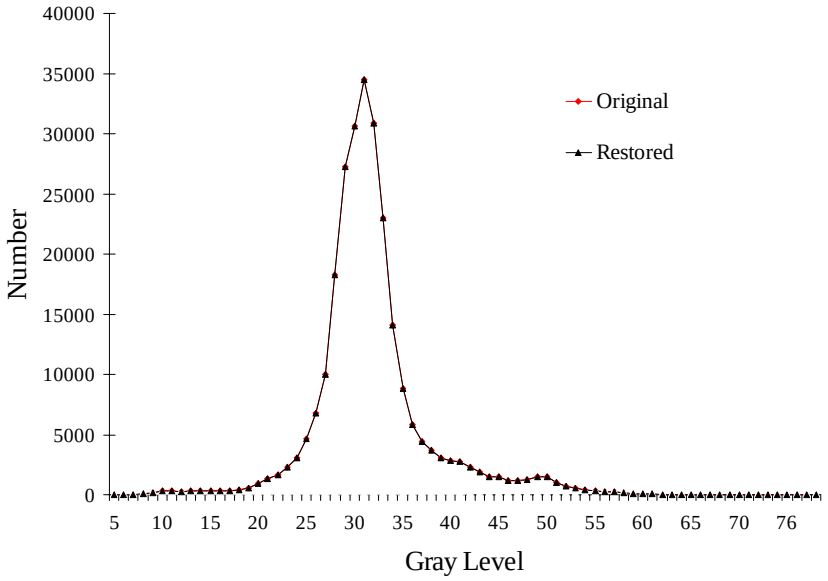


(a)

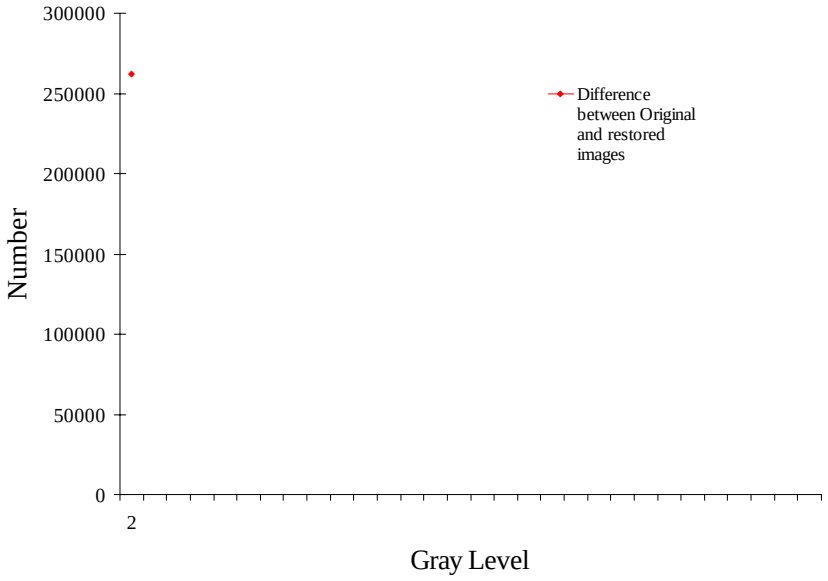


(b)

Figure B.24 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 3

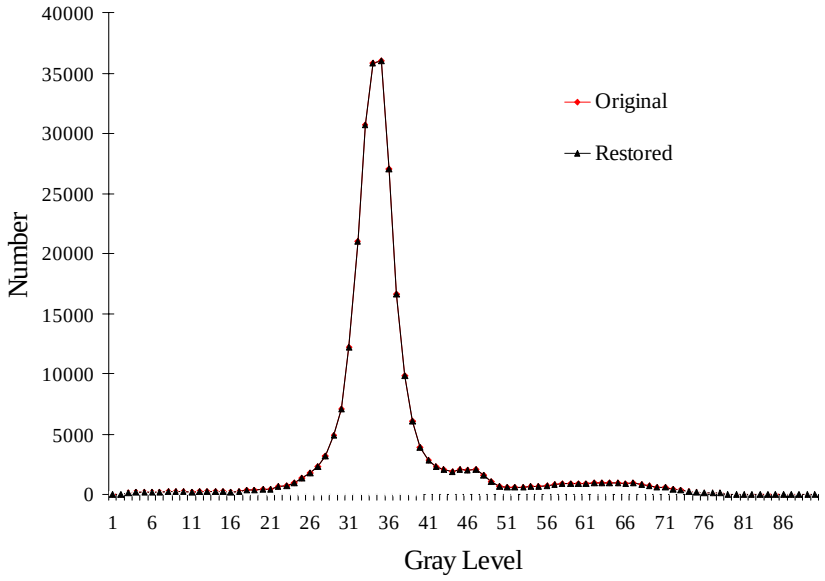


(a)

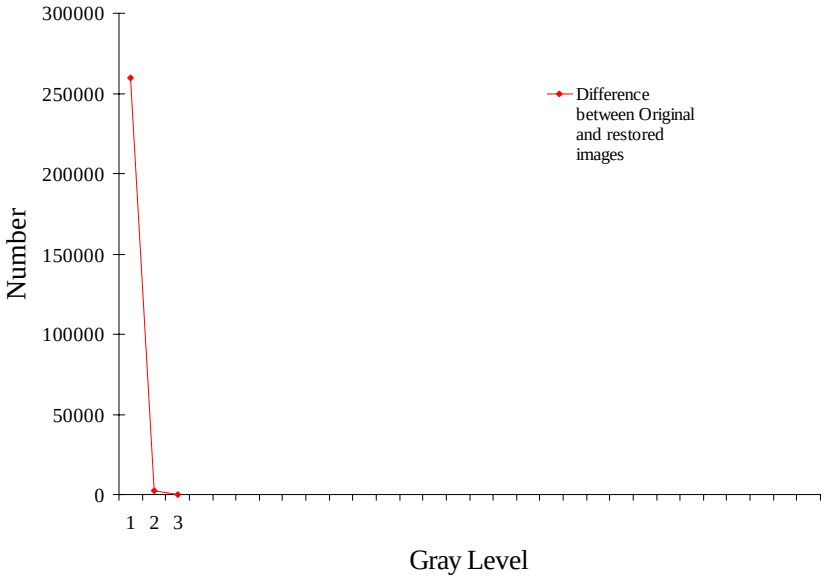


(b)

Figure B.25 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 4

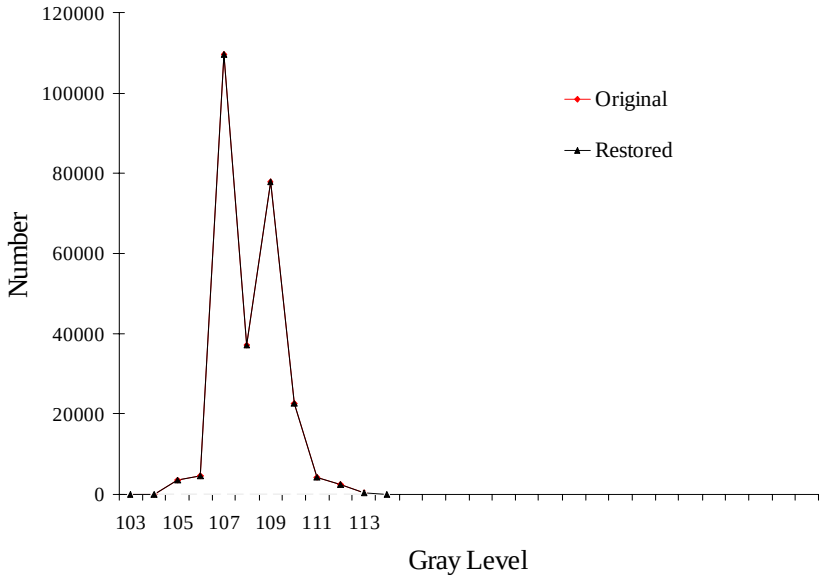


(a)

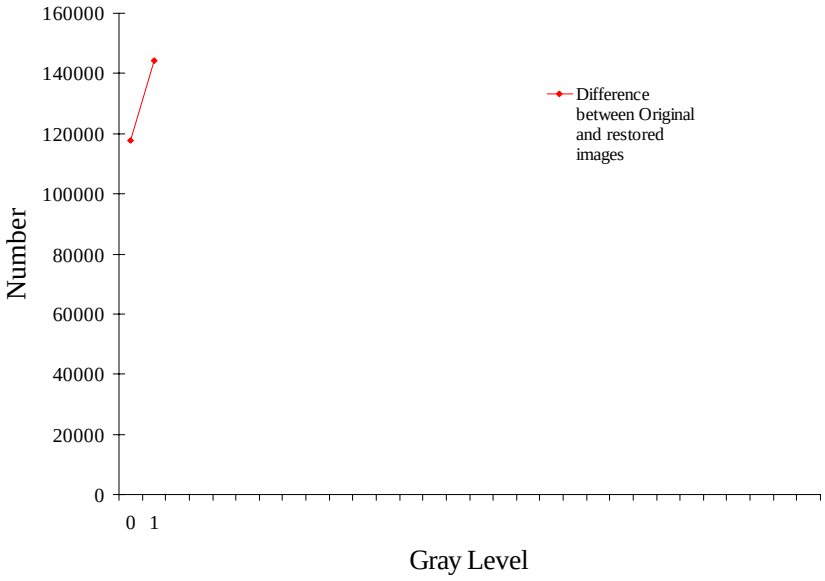


(b)

Figure B.26 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 5

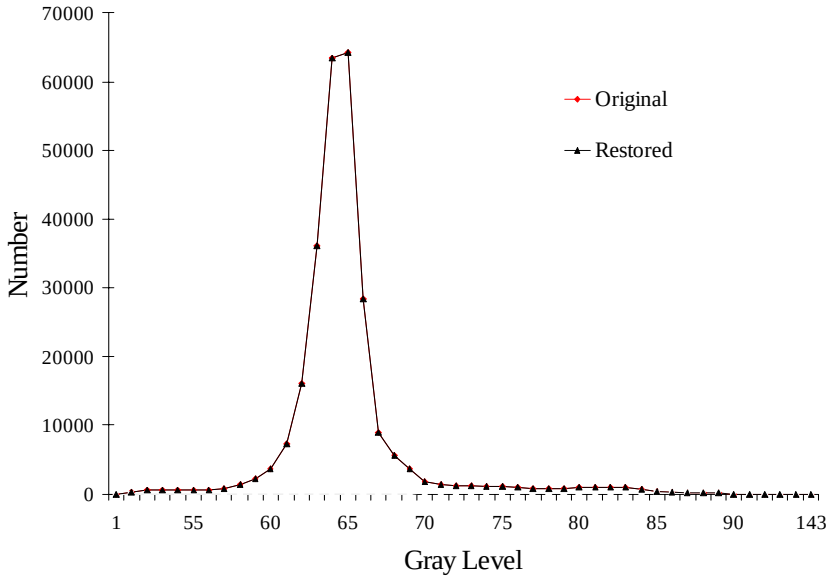


(a)

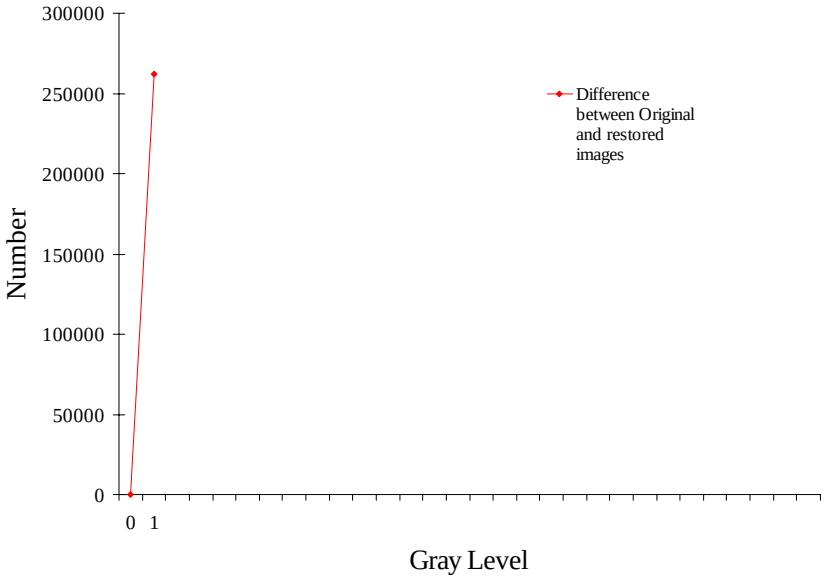


(b)

Figure B.27 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 6

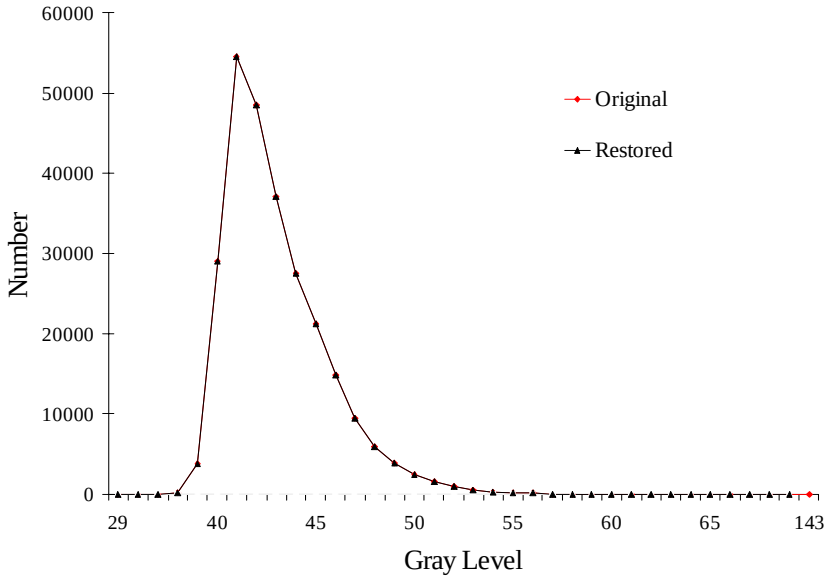


(a)

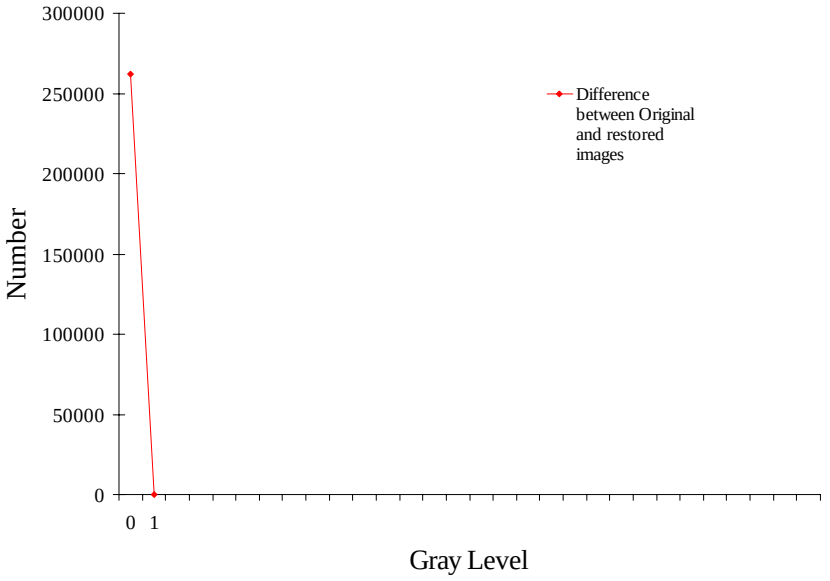


(b)

Figure B.28 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-6, Band 7

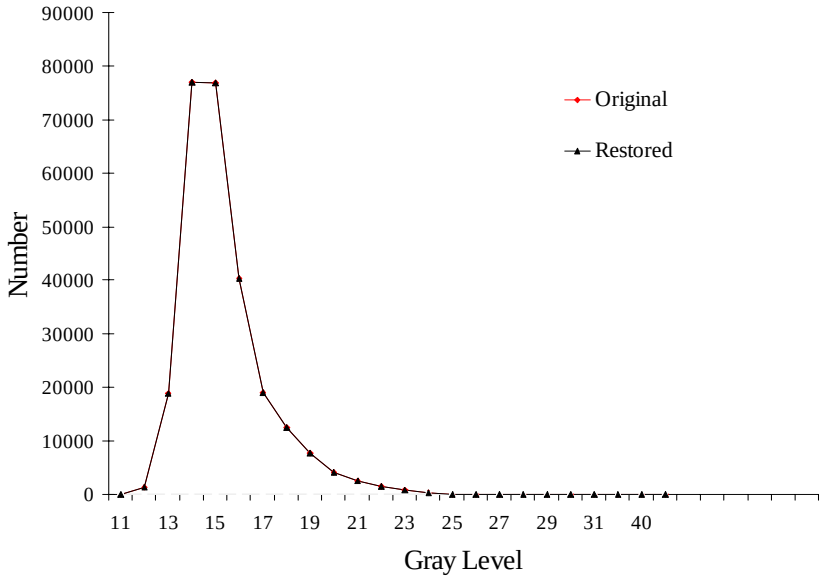


(a)

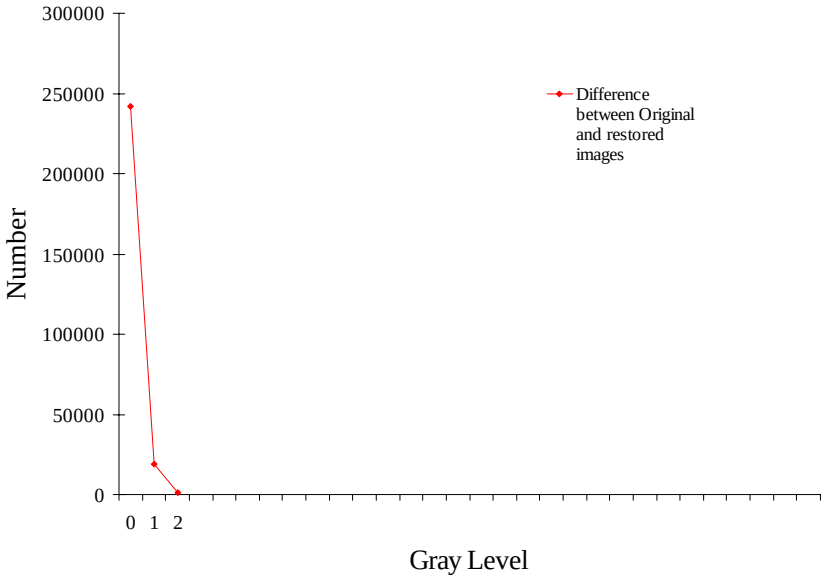


(b)

Figure B.29 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 1

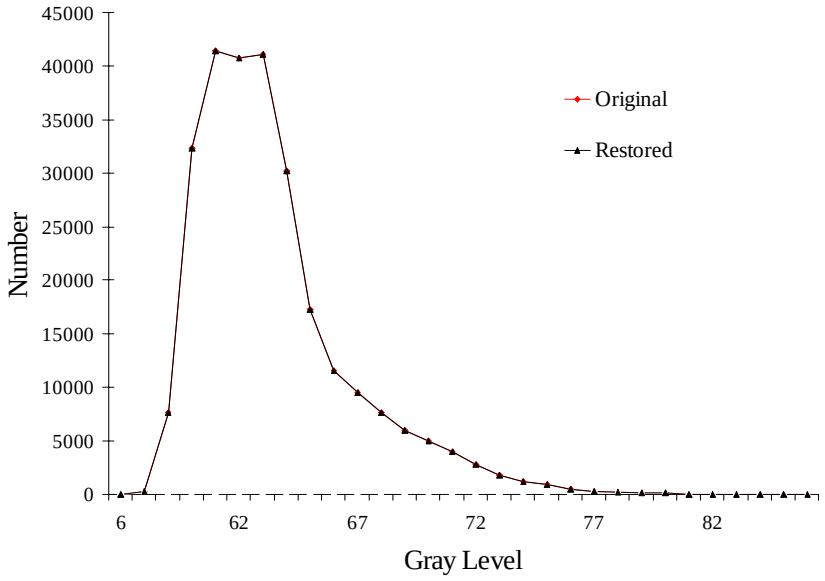


(a)

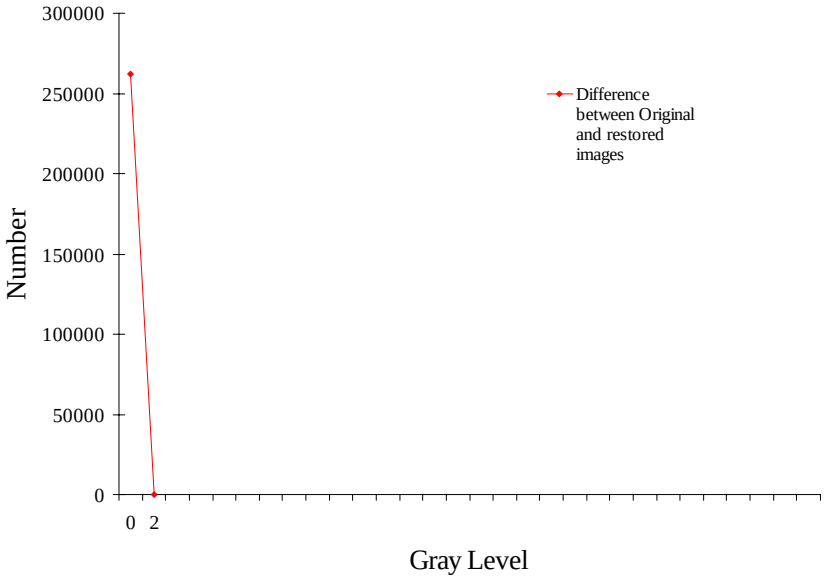


(b)

Figure B.30 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 2

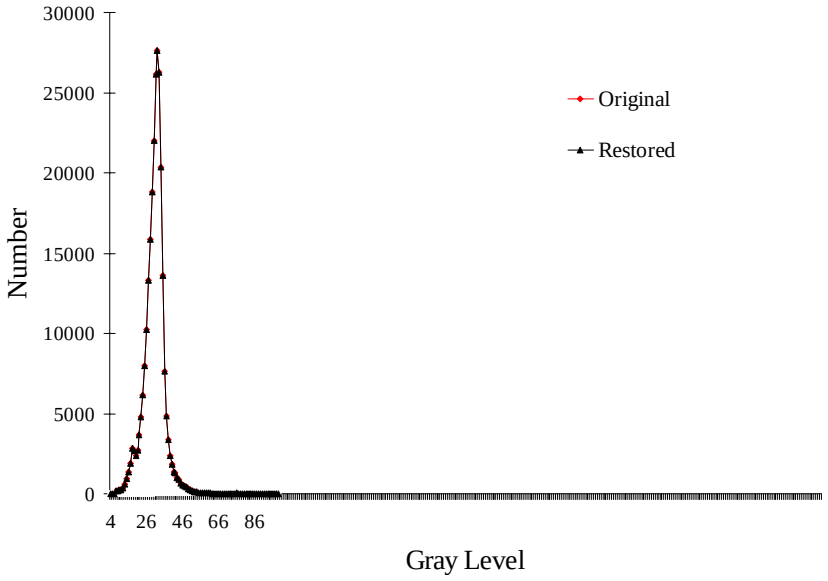


(a)

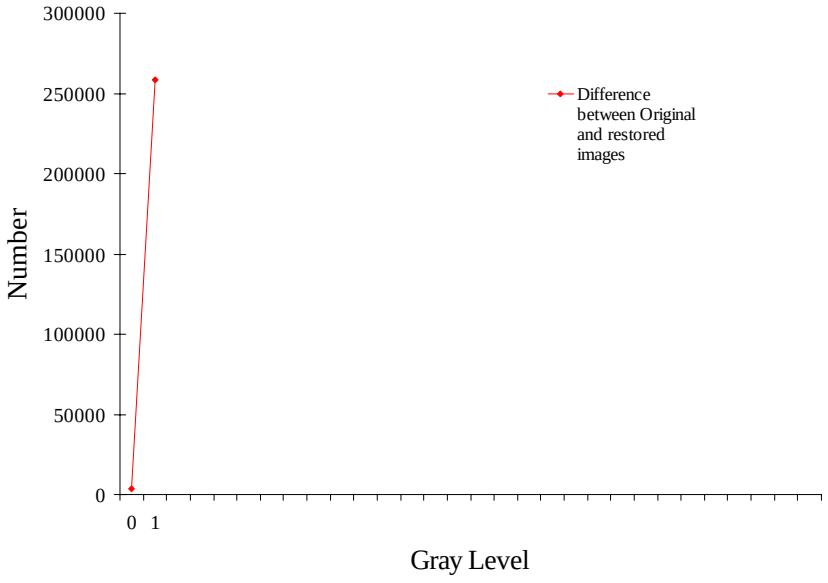


(b)

Figure B.31 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 3

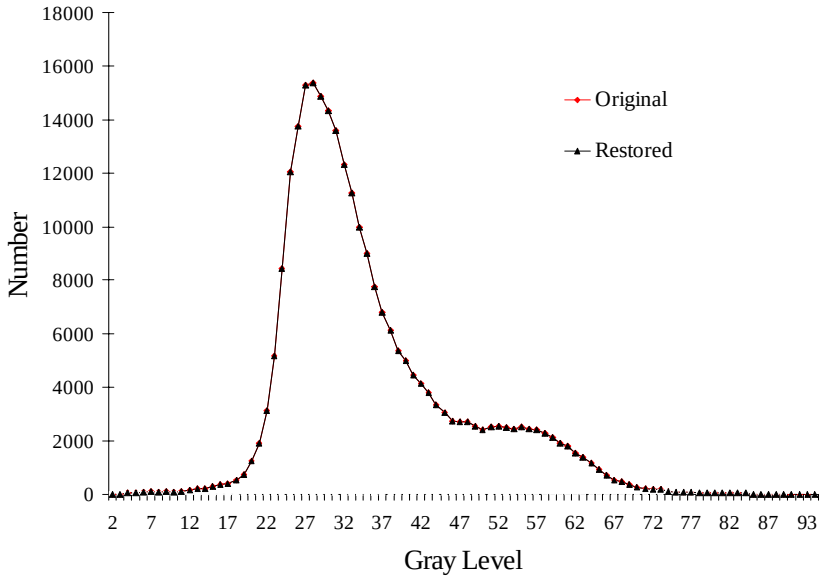


(a)

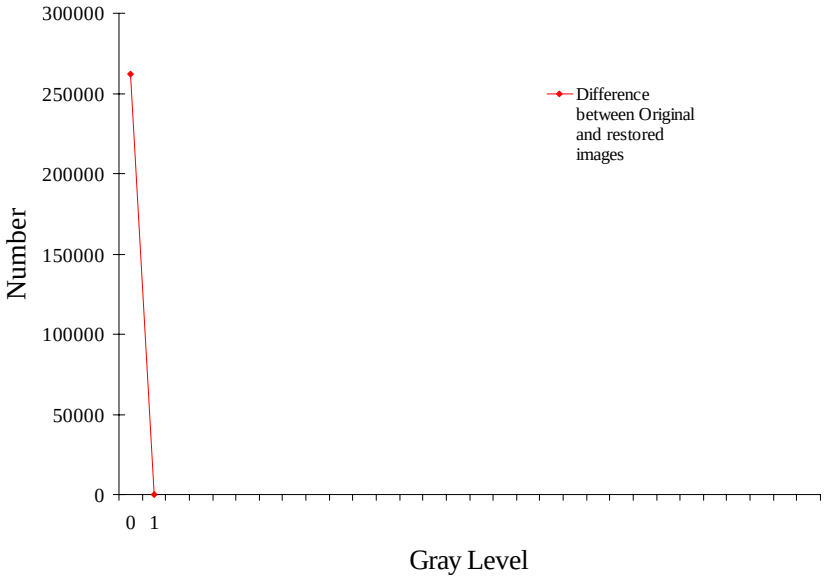


(b)

Figure B.32 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 4

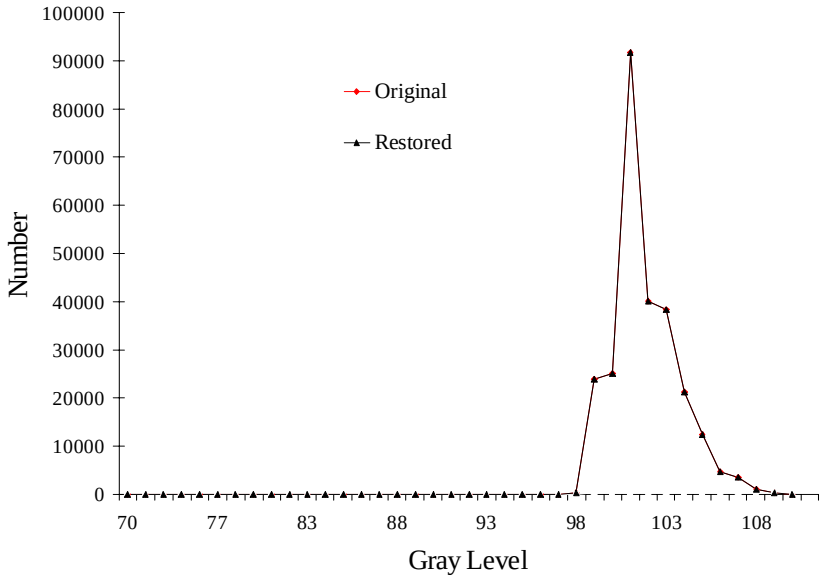


(a)

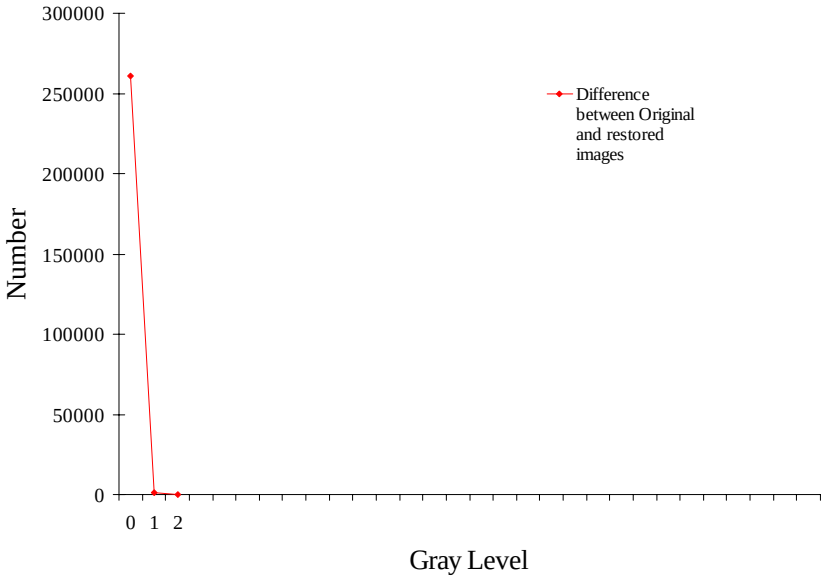


(b)

Figure B.33 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 5

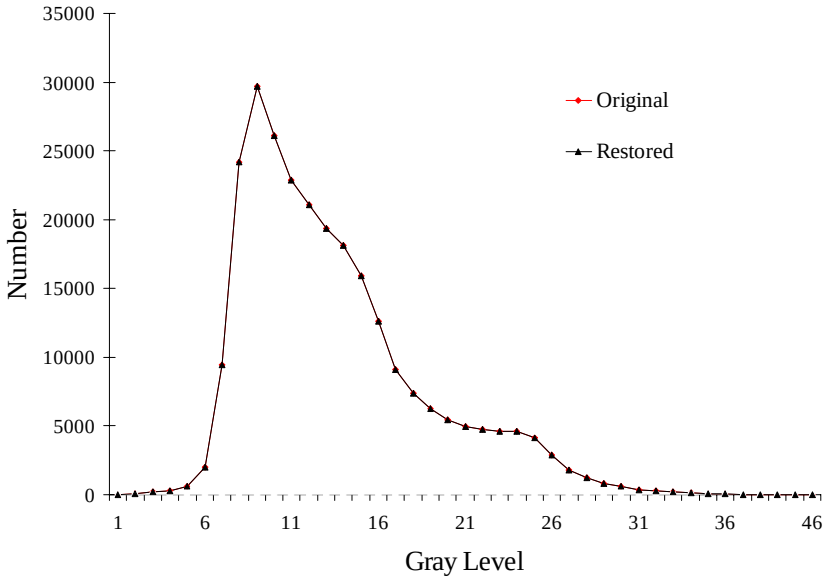


(a)

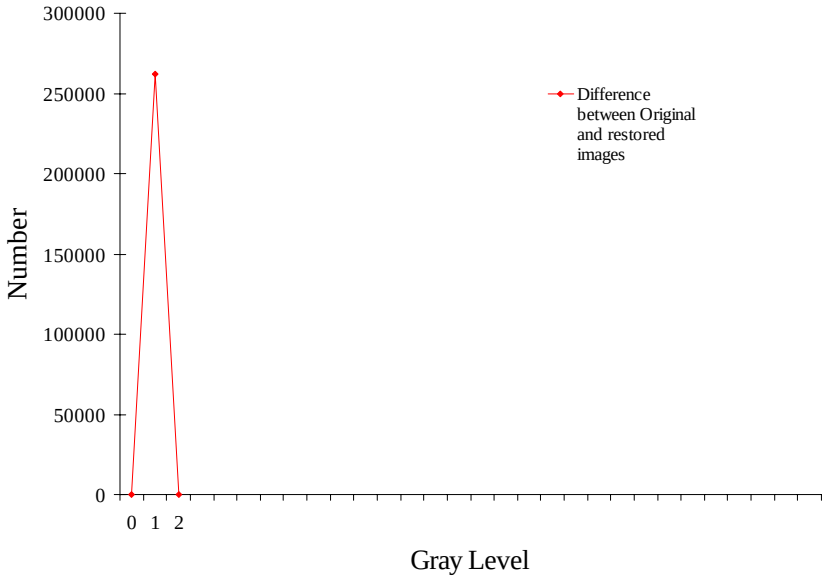


(b)

Figure B.34 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 6

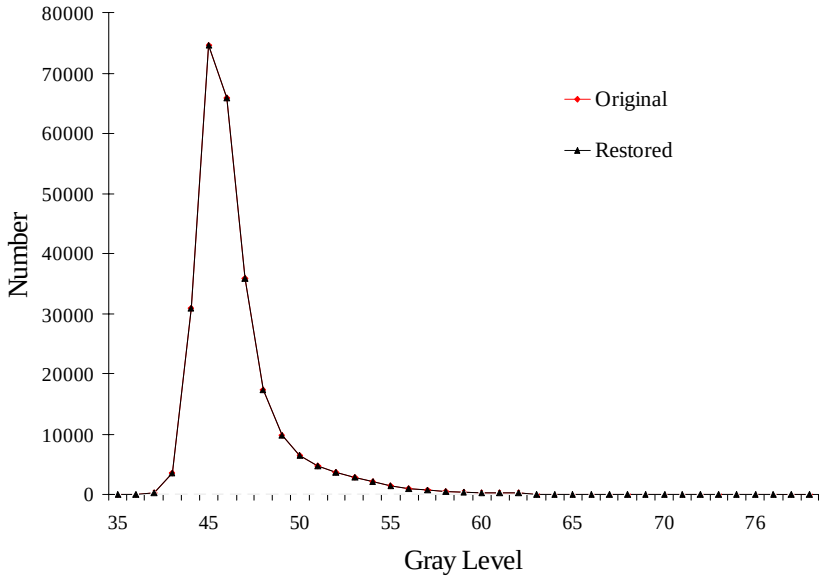


(a)

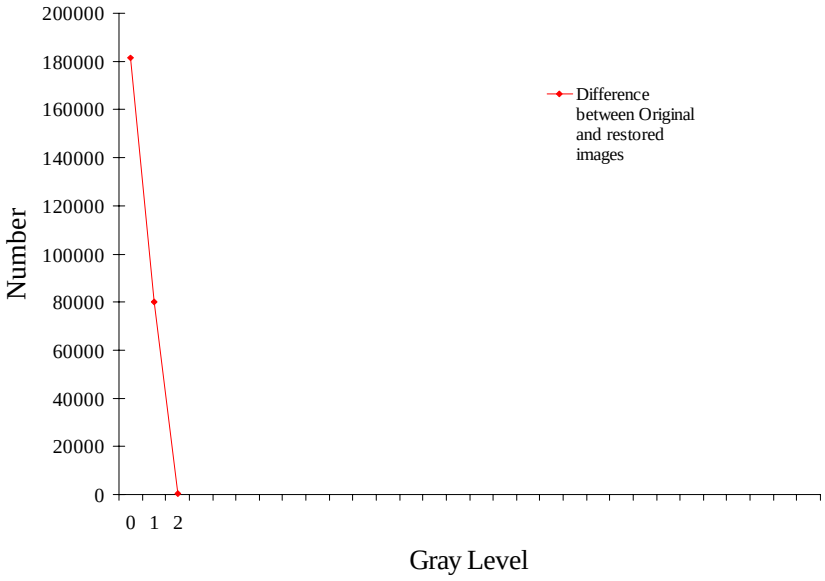


(b)

Figure B.35 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-7, Band 7

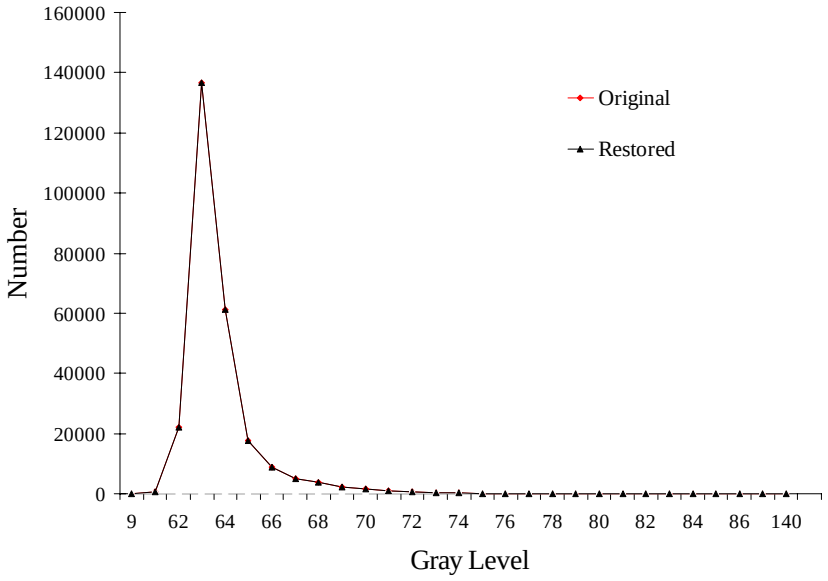


(a)

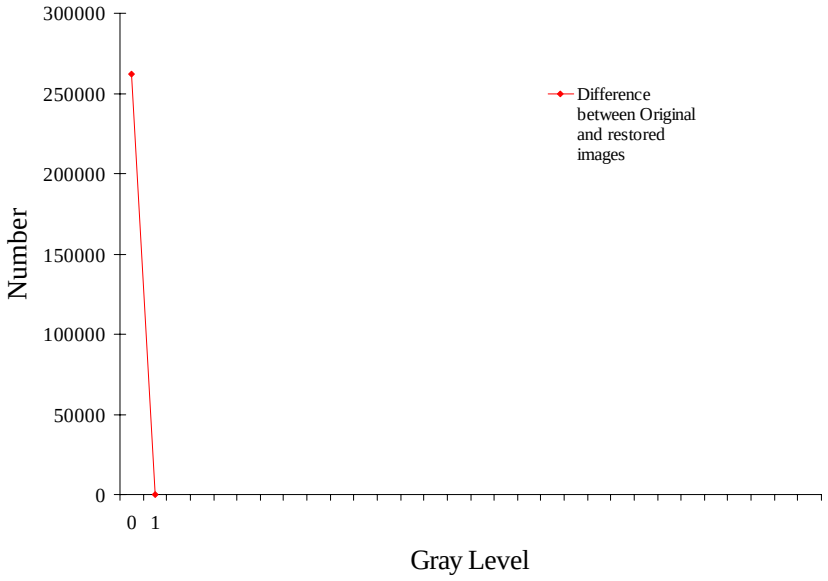


(b)

Figure B.36 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 1

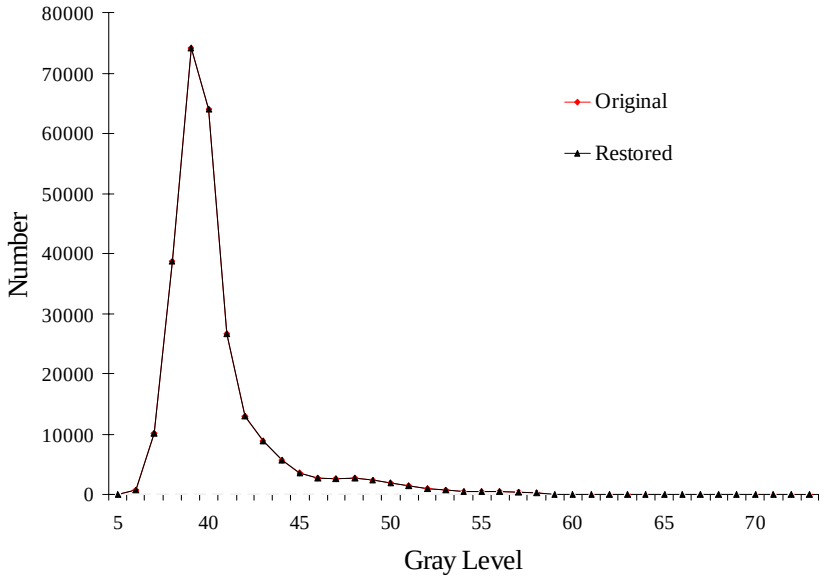


(a)

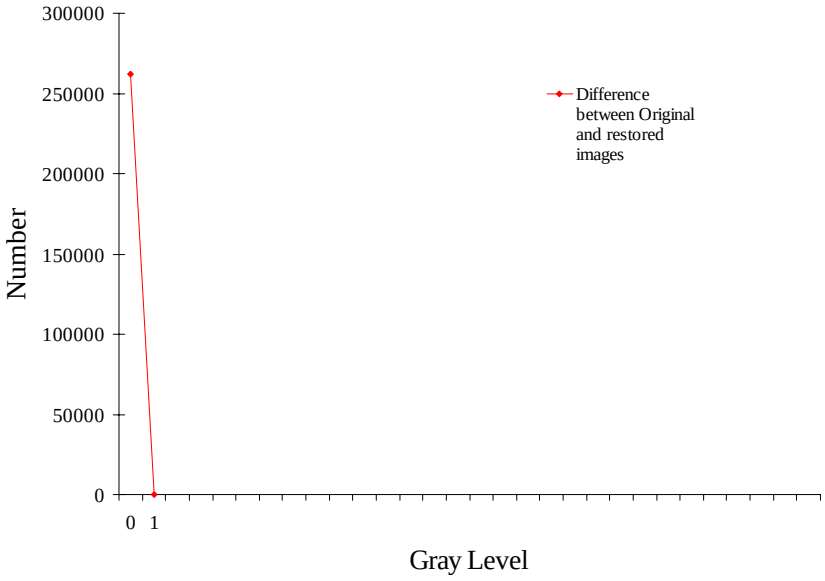


(b)

Figure B.37 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 2

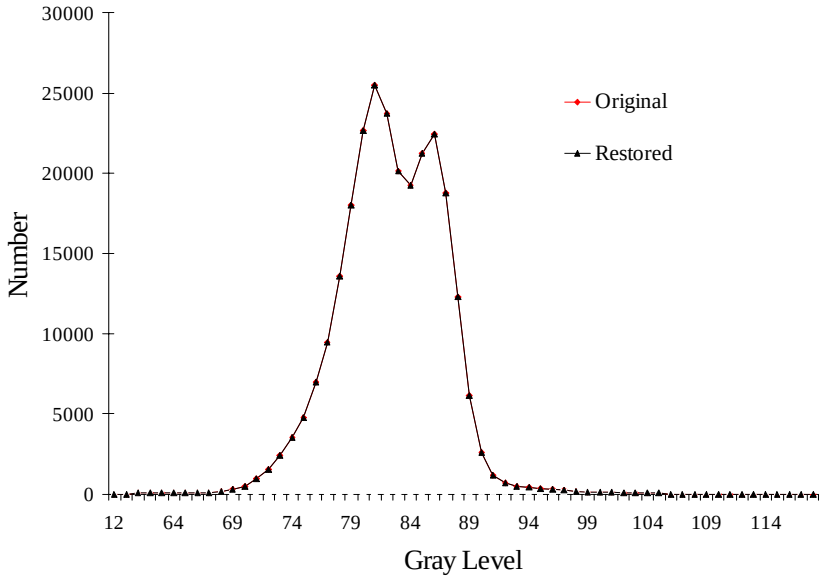


(a)

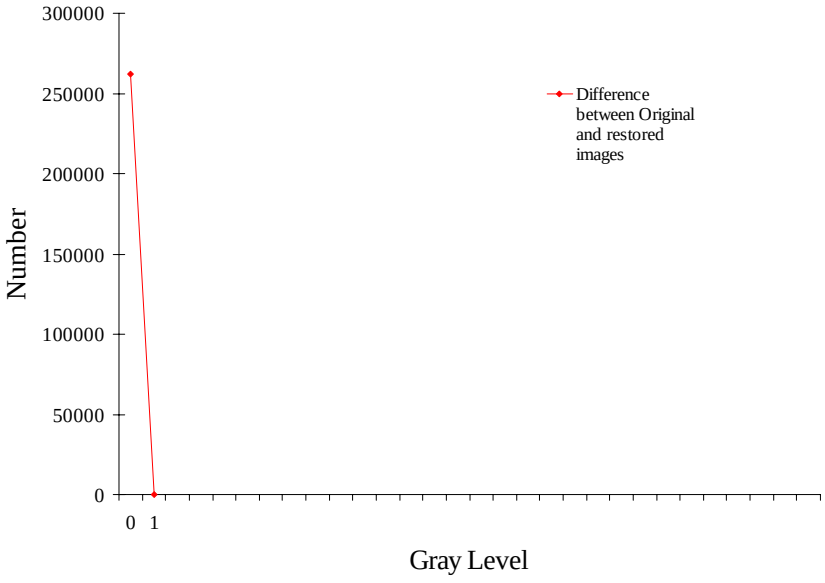


(b)

Figure B.38 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 3

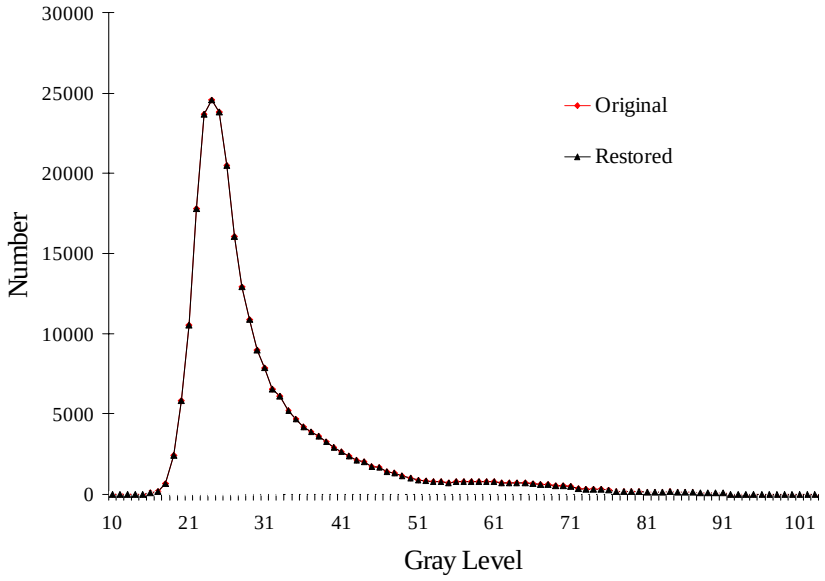


(a)

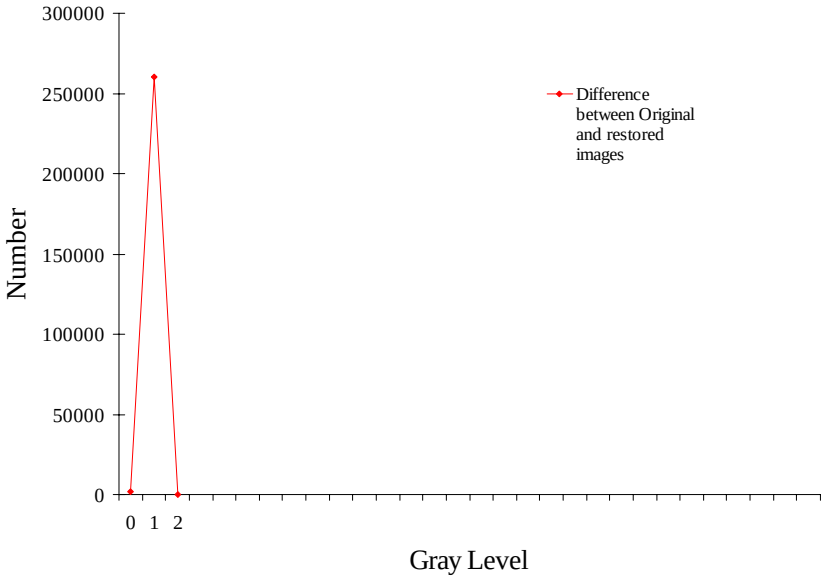


(b)

Figure B.39 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 4

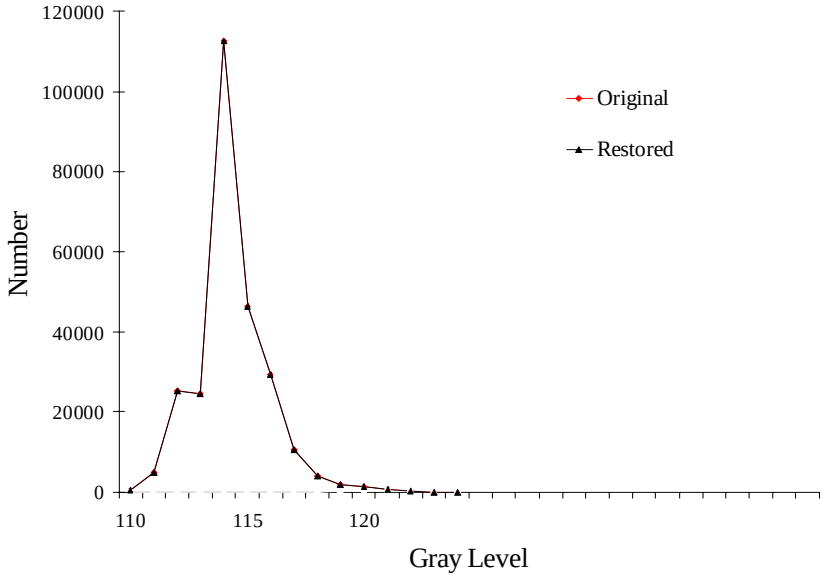


(a)

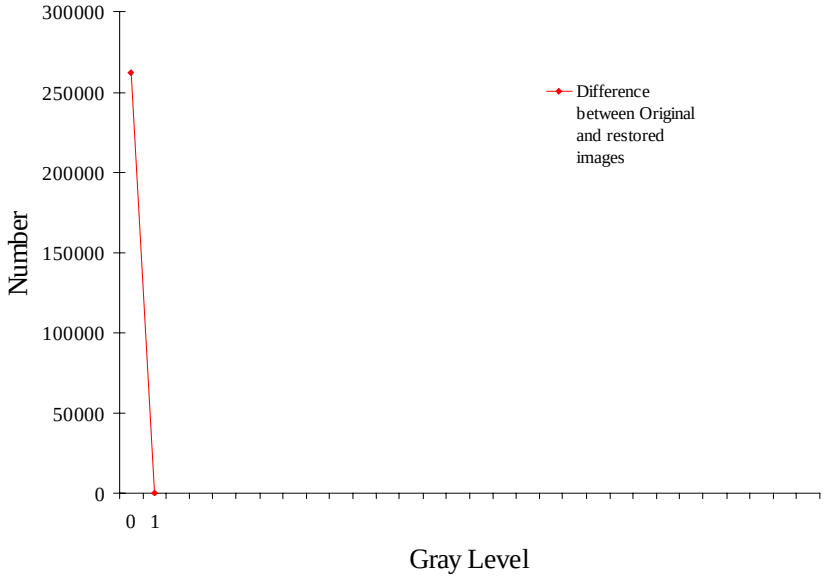


(b)

Figure B.40 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 5

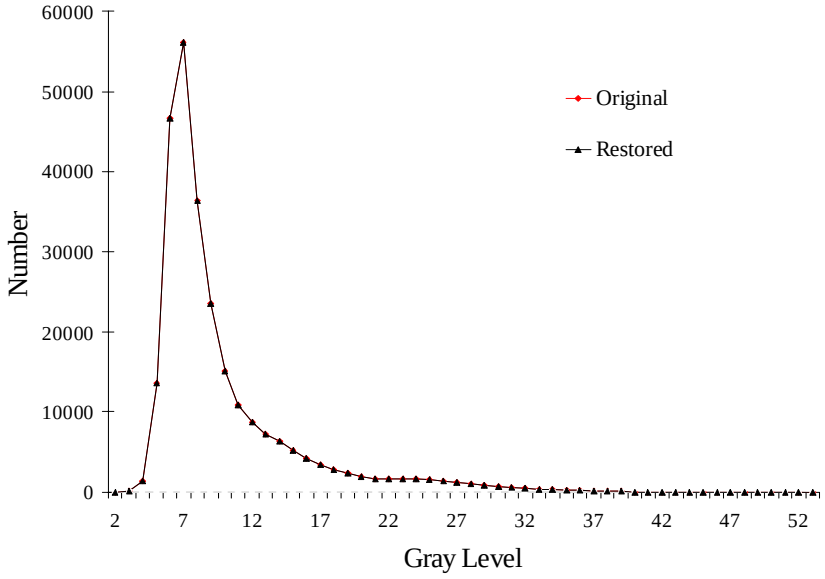


(a)

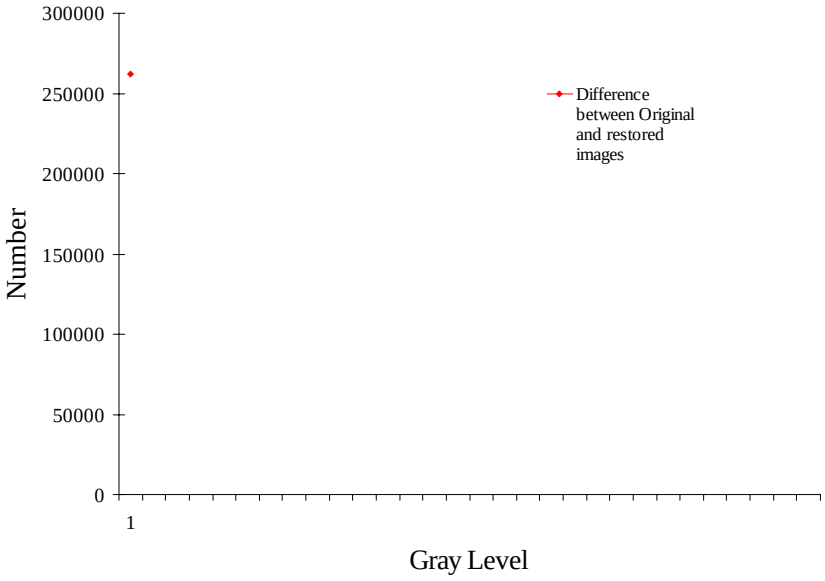


(b)

Figure B.41 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 6



(a)



(b)

Figure B.42 Profiles of DNs in Original and Restored Images (a) and Profiles of DNs in their Difference Image (b), Area-8, Band 7

APPENDIX C

SOURCE CODES

```
/*-------------------------------------------------------------------------------------------------------------*/
/*      Program compressing images using a fractal scheme with a quadtree partition                            */
/*                                                                                                             */
/*      Written by Ke Xiao in 7/1997; modified in 12/2002                                                      */
/*                                                                                                             */
/*      Based on Fisher 1992 and Jacquin 1993                                                                  */
/*-------------------------------------------------------------------------------------------------------------*/

#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <conio.h>
#include <process.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <io.h>

void fatal();
char *strcpy();
void compute_sums();
void quartree_partion();
void quadtree();
int pack();
void list_free();

unsigned char **image;
double     **domimage[4];
double max_scale = 1.0;

int    s_bits, o_bits, min_part, max_part, dom_step,dom_step_type, dom_type;
int    only_positive, subclass_search, fullclass_search, *bits_needed;
int    zero_ialpha, max_exponent;

int    class_transform[2][24], rot_transform[2][8];
struct domain_data {
      int *no_h_domains, *no_v_domains, *domain_hsize, *domain_vsize,
*domain_hstep, *domain_vstep;
struct domain_pixels {
      int dom_x, dom_y, sym;
      double sum,sum2;
      int sym;
} ***pixel;
```

```
} domain;


struct classified_domain {
      struct domain_pixels *the;
      struct classified_domain *next;
} **the_domain[3][24];

FILE *output;
#define bound(a)   ((a) < 0.0 ? 0 : ((a)>255.0? 255 : a))
#define swap(a,b,TYPE)          {TYPE _temp; _temp=b; b=a; a= _temp;}
#define image_array(matrix, hsize, vsize, TYPE) {\
   TYPE *imptr; \
   int _i; \
   matrix = (TYPE **)malloc((vsize)*sizeof(TYPE *));\
   imptr = (TYPE*)malloc((long)(hsize)*(long)(vsize)*sizeof(TYPE));\
   if (imptr == NULL) \
      fatal("\nNo memory in matrix allocate."); \
   for (_i = 0; _i<vsize; ++_i, imptr += hsize) \
      matrix[_i] = imptr; \
 }

main(argc,argv)
int argc;
char **argv;
{
   char          inputfilename[50];
   char          outputfilename[50];
   int           max_value, min_value, i,j,k, hsize, vsize;
   long          stripchar;
   double        tol = 4.0;
   FILE          *input;

   for (i=1; i<argc; ++i)
     if (argv[i][0] != '-' )
         if (inputfilename[0] == 1)
            strcpy(inputfilename, argv[i]);
         else if (outputfilename[0] == 1)
            strcpy(outputfilename, argv[i]);
         else;
     else {
         if (strlen(argv[i]) == 1) break;
         switch(argv[i][1]) {
           case 't': tol = atof(argv[++i]);
                     break;
           case 'S': stripchar = atoi(argv[++i]);
```

```
                break;
        case 'x':
        case 'w': hsize = atoi(argv[++i]);
                break;
        case 'y':
        case 'h': vsize = atoi(argv[++i]);
                break;
        case 'D': dom_type = atoi(argv[++i]);
                break;
        case 'd': dom_step = atoi(argv[++i]);
                if (dom_step < 0 || dom_step > 15)
                  fatal("\n invalid domain step.");
                break;
        case 's': s_bits = atoi(argv[++i]);
                break;
        case 'o': o_bits = atoi(argv[++i]);
                break;
        case 'm': min_part = atoi(argv[++i]);
                break;
        case 'M': max_part = atoi(argv[++i]);
                break;
        case 'e': dom_step_type= 1;
                break;
        case 'p': only_positive = 1;
                break;
        case 'f': subclass_search = 1;
                break;
        case 'F': fullclass_search = 1;
                break;
        case 'N': max_scale = atof(argv[++i]);
                break;
        case '?':
        case 'H':
        default:
                fatal("\n       -F and -f can be used together.");
      }
  }

if (hsize == -1)
   if (vsize == -1) hsize = vsize = 256;
   else hsize = vsize;
else
   if (vsize == -1) vsize = hsize;

image_array(image, hsize, vsize, unsigned char)
image_array(domimage[0], hsize/2, vsize/2, double)
```

```
image_array(domimage[1], hsize/2, vsize/2, double)
image_array(domimage[2], hsize/2, vsize/2, double)
image_array(domimage[3], hsize/2, vsize/2, double)

bits_needed = (int *)malloc(sizeof(int)*(1+max_part-min_part));

if ((input = fopen(inputfilename, "r")) == NULL)
   fatal("Can't open input file.");

fseek(input, stripchar, 0);
i = fread(image[0], sizeof(unsigned char), hsize*vsize, input);
fclose(input);

if (i < hsize*vsize)
   fatal("Invalid input data.");
else
   printf("%dx%d=%d pixels read from %s.", hsize,vsize,i,inputfilename);

max_value=0;
min_value=255;
for(i=0; i<vsize; i++)
   for(j=0; j<hsize; j++) {
      if(image[j][i]<min_value) min_value=image[j][i];
               if(image[j][i]>man_value) max_value=image[j][i];
}

compute_sums(hsize,vsize);

if ((output = fopen(outputfilename, "w")) == NULL)
   fatal("Can't open output file.");

class_transform[2][24] = {23,17,21,11,15,9,22,16,19,5,13,3,20,10,18,
                          4,7,1,14,8,12,2,6,0,
                          16,22,10,20,8,14,17,23,4,18,2,12,11,21,5,
                          19,0,6,9,15,3,13,1,7};
rot_transform[2][8] = {7,4,5,6,1,2,3,0, 2,3,0,1,6,7,4,5};

pack(4,(long)min_part,output);
pack(4,(long)max_part,output);
pack(4,(long)dom_step,output);
pack(1,(long)dom_step_type,output);
pack(2,(long)dom_type,output);
pack(12,(long)hsize,output);
pack(12,(long)vsize,output);
pack(7, (long)max_value);
pack(7, (long)min_value);
```

```
    zero_ialpha = 0.5 + (max_scale)/(2.0*max_scale)*(1<<s_bits);

    printf("\nProcessing.....");
    fflush(stdout);
    quartree_partion(0, 0, hsize,vsize, tol);
    printf("Done.");
    fflush(stdout);

    pack(-1,(long)0,output);

    fclose(output);
    i = pack(-2,(long)0,output);
    printf("\n Compression = %lf from %d bytes written in %s.\n",
          (double)(hsize*vsize)/(double)i, i, outputfilename);

    free(bits_needed);
    free(domimage[0]);
    free(domimage[1]);
    free(domimage[2]);
    free(domimage[3]);
    free(domain.no_h_domains);
    free(domain.no_v_domains);
    free(domain.domain_hsize);
    free(domain.domain_vsize);
    free(domain.domain_hstep);
    free(domain.domain_vstep);
    for (i=0; i <= max_part-min_part; ++i)
       free(domain.pixel[i]);
    free(domain.pixel);
    free(image[0]);
    for (i=0; i <= max_part-min_part; ++i)
    for (k=0; k<3; ++k)
     for (j=0; j<24; ++j) list_free(the_domain[k][j][i]);
    return(0);
}

void list_free(node)
struct classified_domain *node;
{
     if (node->next != NULL)
       list_free(node->next);
     free(node);
}

void average(x,y,xsize,ysize, psum, psum2)
```

```
int x,y,xsize,ysize;
double *psum, *psum2;
{
   register int i,j,k;
   register double pixel;
   *psum = *psum2 = 0.0;
   k = ((x%2)<<1) + y%2;
   x >>= 1; y >>= 1;
   xsize >>= 1; ysize >>= 1;
   for (i=x; i<x+xsize; ++i)
   for (j=y; j<y+ysize; ++j) {
            pixel = domimage[k][j][i];
     *psum += pixel;
     *psum2 += pixel*pixel;
   }
}

void average1(x,y,xsize,ysize, psum, psum2)
int x,y,xsize,ysize;
double *psum, *psum2;
{
   register int i,j;
   register double pixel;
   *psum = *psum2 = 0.0;

   for (i=x; i<x+xsize; ++i)
   for (j=y; j<y+ysize; ++j) {
            pixel = (double)image[j][i];
     *psum += pixel;
     *psum2 += pixel*pixel;
   }
}

classify(x, y, xsize, ysize, pfirst, psecond, psym, psum, psum2, type)
int x,y,xsize,ysize,       *pfirst, *psecond,   *psym
double *psum, *psum2;
int type;
}

   int order[4], i,j;
   double a[4],a2[4];
   void (*average_func)();

   if (type == 2) average_func = average; else average_func = average1;

   (*average_func)(x,y,                xsize/2,ysize/2,   &a[0], &a2[0]);
```

```
(*average_func)(x,y+ysize/2,         xsize/2,ysize/2,   &a[1], &a2[1]);
(*average_func)(x+xsize/2,y+ysize/2, xsize/2,ysize/2,   &a[2], &a2[2]);
(*average_func)(x+xsize/2,y,         xsize/2,ysize/2,   &a[3], &a2[3]);

*psum = a[0] + a[1] + a[2] + a[3];
*psum2 =  a2[0] + a2[1] + a2[2] + a2[3];

for (i=0; i<4; ++i) {
    order[i] = i;
    a2[i] -= (double)(1<<(2*type))*a[i]*a[i]/(double)(xsize*ysize);
}

for (i=2; i>=0; --i)
for (j=0; j<=i; ++j)
   if (a[j]<a[j+1]) {
       swap(order[j], order[j+1],int)
       swap(a[j], a[j+1],double)
}

*psym = order[0];
for (i=0; i<4; ++i)
   order[i] = (order[i] - (*psym) + 4)%4;

for (i=0; order[i] != 2; ++i);
*pfirst = i-1;
if (order[3] == 1 || (*pfirst == 2 && order[2] == 1)) *psym += 4;

for (i=0; i<4; ++i) order[i] = i;

for (i=2; i>=0; --i)
for (j=0; j<=i; ++j)
   if (a2[j]<a2[j+1]) {
       swap(order[j], order[j+1],int)
       swap(a2[j], a2[j+1],double)
}

for (i=0; i<4; ++i)
   order[i] = (order[i] - (*psym%4) + 4)%4;
if (*psym > 3)
   for (i=0; i<4; ++i)
     if (order[i]%2) order[i] = (2 + order[i])%4;

*psecond = 0;
for (i=2; i>=0; --i)
for (j=0; j<=i; ++j)
   if (order[j] > order[j+1]) {
```

```
            swap(order[j],order[j+1], int);
            if (order[j] == 0 || order [j+1] == 0)
                *psecond += 6;
            else if (order[j] == 1 || order [j+1] == 1)
                *psecond += 2;
            else if (order[j] == 2 || order [j+1] == 2)
                *psecond += 1;
        }
}

void compute_sums(hsize,vsize)
int hsize,vsize;
{
    int i,j,k,l, domain_x, domain_y, first_class, second_class, size,  x_exponent, y_exponent;

    struct classified_domain *node;

    printf("\nComputing domain sums... ");
    fflush(stdout);

    for (i=0; i<2; ++i)
    for (j=0; j<2; ++j)
    for (k=i; k<hsize-i; k += 2)
    for (l=j; l<vsize-j; l += 2)
        domimage[(i<<1)+j][l>>1][k>>1] =
                ((double)image[l][k] + (double)image[l+1][k+1] +
                (double)image[l][k+1] + (double)image[l+1][k])*0.25;

    x_exponent = (int)floor(log((double)hsize)/log(2.0));
    y_exponent = (int)floor(log((double)vsize)/log(2.0));

    max_exponent = (x_exponent > y_exponent ? y_exponent : x_exponent);

    size =  1<<max_exponent;

    if (max_exponent < max_part)
      fatal("Reduce maximum number of quadtree partitions.");
    if (max_exponent-2 < max_part)
      printf("\nWarning: so many quadtree partitions yield absurd ranges.");

    i = max_part - min_part + 1;
    domain.no_h_domains = (int *)malloc(sizeof(int)*i);
    domain.no_v_domains = (int *)malloc(sizeof(int)*i);
    domain.domain_hsize = (int *)malloc(sizeof(int)*i);
    domain.domain_vsize = (int *)malloc(sizeof(int)*i);
    domain.domain_hstep = (int *)malloc(sizeof(int)*i);
```

```
domain.domain_vstep = (int *)malloc(sizeof(int)*i);

domain.pixel= (struct domain_pixels ***)
        malloc(i*sizeof(struct domain_pixels **));
if (domain.pixel == NULL) fatal("No memory for domain pixel sums.");

for (i=0; i <= max_part-min_part; ++i) {
  domain.domain_hsize[i] = size >> (min_part+i-1);
  if (dom_type == 2)
       domain.domain_hstep[i] = dom_step;
  else if (dom_type == 1)
       if (dom_step_type == 1)
         domain.domain_hstep[i] = (size >> (max_part - i-1))*dom_step;
       else
         domain.domain_hstep[i] = (size >> (max_part - i-1))/dom_step;
  else
       if (dom_step_type == 1)
         domain.domain_hstep[i] = domain.domain_hsize[i]*dom_step;
       else
         domain.domain_hstep[i] = domain.domain_hsize[i]/dom_step;

  domain.no_h_domains[i] = 1+(hsize-domain.domain_hsize[i])/
                                  domain.domain_hstep[i];
  domain.domain_vsize[i] = size >> (min_part+i-1);
  if (dom_type == 2)
       domain.domain_vstep[i] = dom_step;
  else if (dom_type == 1)
       if (dom_step_type == 1)
         domain.domain_vstep[i] = (size >> (max_part - i-1))*dom_step;
       else
         domain.domain_vstep[i] = (size >> (max_part - i-1))/dom_step;
  else
       if (dom_step_type == 1)
         domain.domain_vstep[i] = domain.domain_vsize[i]*dom_step;
       else
         domain.domain_vstep[i] = domain.domain_vsize[i]/dom_step;

  domain.no_v_domains[i] = 1+(vsize-domain.domain_vsize[i])/
                                  domain.domain_vstep[i];

  bits_needed[i] = ceil(log((double)domain.no_h_domains[i]*
                    (double)domain.no_v_domains[i])/log(2.0));

  image_array(domain.pixel[i], domain.no_h_domains[i],
             domain.no_v_domains[i], struct domain_pixels)
}
```

```
i = max_part - min_part + 1;
for (first_class = 0; first_class < 3; ++first_class)
for (second_class = 0; second_class < 24; ++second_class) {
  the_domain[first_class][second_class] =
                  (struct classified_domain **)
                  malloc(i*sizeof(struct classified_domain *));
  for (j=0; j<i; ++j)
            the_domain[first_class][second_class][j] = NULL;
}

for (i=0; i <= max_part-min_part; ++i) {
 for (j=0,domain_x=0; j<domain.no_h_domains[i]; ++j,
       domain_x+=domain.domain_hstep[i])
 for (k=0,domain_y=0; k<domain.no_v_domains[i]; ++k,
       domain_y+=domain.domain_vstep[i]) {
              classify(domain_x, domain_y,
          domain.domain_hsize[i],
          domain.domain_vsize[i],
                                    &first_class, &second_class,
                                    &domain.pixel[i][k][j].sym,
            &domain.pixel[i][k][j].sum,
            &domain.pixel[i][k][j].sum2, 2);

   domain.pixel[i][k][j].dom_x = j;
   domain.pixel[i][k][j].dom_y = k;
   node = (struct classified_domain *)
                    malloc(sizeof(struct classified_domain));

   node->the = &domain.pixel[i][k][j];
   node->next = the_domain[first_class][second_class][i];
   the_domain[first_class][second_class][i] = node;
        }
}

for (i=0; i <= max_part-min_part; ++i)
for (first_class = 0; first_class < 3; ++first_class)
for (second_class = 0; second_class < 24; ++second_class)
  if (the_domain[first_class][second_class][i] == NULL) {
     node = (struct classified_domain *)
                    malloc(sizeof(struct classified_domain));
     node->the = &domain.pixel[i][0][0];
     node->next = NULL;
     the_domain[first_class][second_class][i] = node;
  }
```

```
    printf("Done.");
    fflush(stdout);
}

int pack(size, value, foutf)
int size; long int value;
FILE *foutf;
{
     int i;
     static int ptr = 1, sum = 0, num_of_packed_bytes = 0;

    if (size == -1 && ptr != 1) {
        fputc(sum<<(8-ptr), foutf);
        ++num_of_packed_bytes;
        return(0);
    }

    if (size == -2)
            return(num_of_packed_bytes);

    for (i=0; i<size; ++i, ++ptr, value = value>>1, sum = sum<<1) {
        if (value & 1) sum |= 1;

         if (ptr == 8) {
            fputc(sum,foutf);
            ++num_of_packed_bytes;
            sum=0;
            ptr=0;
        }
     }
}

double compare(atx,aty, xsize, ysize, depth, rsum, rsum2, dom_x,dom_y,
                              sym_op, pialpha,pibeta)
int atx, aty, xsize, ysize, depth, dom_x, dom_y, sym_op, *pialpha, *pibeta;
double rsum, rsum2;
{
    int i, j, i1, j1, k, domain_x, domain_y;

    double pixel, det,  dsum,  rdsum = 0,
           dsum2,   w2 = 0,   rms = 0,  alpha,   beta;

   w2 = xsize * ysize;

    dsum = domain.pixel[depth-min_part][dom_y][dom_x].sum;
    dsum2 = domain.pixel[depth-min_part][dom_y][dom_x].sum2;
```

```
domain_x = (dom_x * domain.domain_hstep[depth-min_part]);
domain_y = (dom_y * domain.domain_vstep[depth-min_part]);
k = ((domain_x%2)<<1) + domain_y%2;
domain_x >>= 1;
domain_y >>= 1;

#define COMPUTE_LOOP            {                           \
   pixel = domimage[k][j1][i1];                             \
   rdsum += image[j][i]*pixel;                              \
   }

 switch(sym_op) {
    case 0: for (i=atx, i1 = domain_x; i<atx+xsize; ++i, ++i1)
            for (j=aty, j1 = domain_y; j<aty+ysize; ++j, ++j1)
               COMPUTE_LOOP
            break;
    case 1: for (j=aty, i1 = domain_x; j<aty+ysize; ++j, ++i1)
            for (i=atx+xsize-1, j1 = domain_y; i>=atx; --i, ++j1)
               COMPUTE_LOOP
            break;
    case 2: for (i=atx+xsize-1, i1 = domain_x; i>=atx; --i, ++i1)
            for (j=aty+ysize-1, j1 = domain_y; j>=aty; --j, ++j1)
               COMPUTE_LOOP
            break;
    case 3: for (j=aty+ysize-1, i1 = domain_x; j>=aty; --j, ++i1)
            for (i=atx, j1 = domain_y; i<atx+xsize; ++i, ++j1)
               COMPUTE_LOOP
            break;
    case 4: for (j=aty, i1 = domain_x; j<aty+ysize; ++j, ++i1)
            for (i=atx, j1 = domain_y; i<atx+xsize; ++i, ++j1)
               COMPUTE_LOOP
            break;
    case 5: for (i=atx, i1 = domain_x; i<atx+xsize; ++i, ++i1)
            for (j=aty+ysize-1, j1 = domain_y; j>=aty; --j, ++j1)
               COMPUTE_LOOP
            break;
    case 6: for (j=aty+ysize-1, i1 = domain_x; j>=aty; --j, ++i1)
            for (i=atx+xsize-1, j1 = domain_y; i>=atx; --i, ++j1)
               COMPUTE_LOOP
            break;
    case 7: for (i=atx+xsize-1, i1 = domain_x; i>=atx; --i, ++i1)
            for (j=aty, j1 = domain_y; j<aty+ysize; ++j, ++j1)
               COMPUTE_LOOP
            break;
 }
```

```
    det = (xsize*ysize)*dsum2 - dsum*dsum;

    if (det == 0.0)
       alpha = 0.0;
     else
       alpha = (w2*rdsum - rsum*dsum)/det;

    if  (only_positive && alpha < 0.0) alpha = 0.0;
    *pialpha = 0.5 + (alpha + max_scale)/(2.0*max_scale)*(1<<s_bits);
    if (*pialpha < 0) *pialpha = 0;
    if (*pialpha >= 1<<s_bits) *pialpha = (1<<s_bits)-1;

    alpha = (double)*pialpha/(double)(1<<s_bits)*(2.0*max_scale)-max_scale;

    beta = (rsum - alpha*dsum)/w2;

    if (alpha > 0.0)  beta += alpha*255;
    *pibeta = 0.5 + beta/
          ((1.0+fabs(alpha))*255)*((1<<o_bits)-1);
    if (*pibeta< 0) *pibeta = 0;
    if (*pibeta>= 1<<o_bits) *pibeta = (1<<o_bits)-1;

    beta = (double)*pibeta/(double)((1<<o_bits)-1)*
            ((1.0+fabs(alpha))*255);
    if (alpha > 0.0) beta  -= alpha*255;

    rms= sqrt((rsum2 + alpha*(alpha*dsum2 - 2.0*rdsum + 2.0*beta*dsum) +
              beta*(beta*w2 - 2.0*rsum))/w2);

    return(rms);
}

void quadtree(atx,aty,xsize,ysize,tol,depth)
int atx, aty, xsize, ysize, depth;
double tol;
{
   int i, sym_op, ialpha, ibeta, best_ialpha, best_ibeta, best_sym_op,
      best_domain_x, best_domain_y, first_class, the_first_class,
      first_class_start, first_class_end, second_class[2],
      the_second_class, second_class_start, second_class_end,
      range_sym_op[2], domain_sym_op;

   struct classified_domain *node;
   double rms, best_rms,
         sum=0, sum2=0;
```

```
if (depth < min_part) {
    quadtree(atx,aty, xsize/2, ysize/2, tol,depth+1);
    quadtree(atx+xsize/2,aty, xsize/2, ysize/2,tol,depth+1);
    quadtree(atx,aty+ysize/2, xsize/2, ysize/2,tol,depth+1);
    quadtree(atx+xsize/2,aty+ysize/2, xsize/2, ysize/2,tol,depth+1);
    return;
}

best_rms = 10000.0;

classify(atx, aty, xsize,ysize,
        &the_first_class, &the_second_class,
        &range_sym_op[0], &sum, &sum2, 1);


if (fullclass_search) {
    first_class_start = 0;
    first_class_end = 3;
} else {
    first_class_start = the_first_class;
    first_class_end = the_first_class+1;
}

if (subclass_search) {
    second_class_start = 0;
    second_class_end = 24;
} else {
    second_class_start = the_second_class;
    second_class_end = the_second_class+1;
}

for (first_class = first_class_start;
    first_class < first_class_end; ++first_class)
for (second_class[0] = second_class_start;
    second_class[0] < second_class_end; ++second_class[0]) {


  if (!only_positive) {
     second_class[1] =
             class_transform[(first_class == 2 ? 1 : 0)][second_class[0]];
     range_sym_op[1] =
        rot_transform[(the_first_class == 2 ? 1 : 0)][range_sym_op[0]];
  }

  for (i=0; i<(2-only_positive); ++i)
```

```
      for (node = the_domain[first_class][second_class[i]][depth-min_part];
         node != NULL;
         node = node->next) {
         domain_sym_op = node->the->sym;
         if (((domain_sym_op>3 ? 4: 0) + (range_sym_op[i]>3 ? 4: 0))%8 == 0)
           sym_op = (4 + domain_sym_op%4 - range_sym_op[i]%4)%4;
         else
           sym_op = (4 + (domain_sym_op%4 + 3*(4-range_sym_op[i]%4))%4)%8;

         rms = compare(atx,aty, xsize, ysize,  depth, sum,sum2,
                        node->the->dom_x,
                        node->the->dom_y,
                        sym_op, &ialpha,&ibeta);

         if (rms < best_rms) {
               best_ialpha = ialpha;
               best_ibeta = ibeta;
               best_rms = rms;
               best_sym_op = sym_op;
               best_domain_x = node->the->dom_x;
               best_domain_y = node->the->dom_y;
         }
      }
   }

   if (best_rms > tol && depth < max_part) {
      pack(1,(long)1,output);
      quadtree(atx,aty, xsize/2, ysize/2, tol,depth+1);
      quadtree(atx+xsize/2,aty, xsize/2, ysize/2, tol,depth+1);
      quadtree(atx,aty+ysize/2, xsize/2, ysize/2, tol,depth+1);
      quadtree(atx+xsize/2,aty+ysize/2, xsize/2, ysize/2, tol,depth+1);
   } else {
       if (depth < max_part)
            pack(1,(long)0,output);
      pack(s_bits, (long)best_ialpha, output);
      pack(o_bits, (long)best_ibeta, output);

      if (best_ialpha != zero_ialpha) {
        pack(3, (long)best_sym_op, output);
        pack(bits_needed[depth-min_part], (long)(best_domain_y*
          domain.no_h_domains[depth-min_part]+best_domain_x), output);
      }
     }
}

void quartree_partion(atx, aty, hsize,vsize, tol)
```

```
int atx, aty, hsize,vsize;
double tol;
{
  int x_exponent,
      y_exponent,
      exponent,
      size,
      depth;

  x_exponent = (int)floor(log((double)hsize)/log(2.0));
  y_exponent = (int)floor(log((double)vsize)/log(2.0));

  exponent = (x_exponent > y_exponent ? y_exponent : x_exponent);
  size = 1<<exponent;
  depth = max_exponent - exponent;
  quadtree(atx,aty,size,size,tol,depth);
  if (size != hsize)
    quartree_partion(atx+size, aty, hsize-size,vsize, tol);

  if (size != vsize)
    quartree_partion(atx, aty+size, size,vsize-size, tol);
}

void fatal(s)
char *s;
{
    printf("%s\n",s);
    exit(-1);
}
```

```
/*-------------------------------------------------------------------------------------------------------------*/
/*        Program decompressing images using a fractal scheme with a quadtree partition                       */
/*                                                                                                            */
/*        Written by Ke Xiao in 7/1997; modified in 12/2002                                                   */
/*                                                                                                            */
/*        Based on Fisher 1992 and Jacquin 1993                                                               */
/*-------------------------------------------------------------------------------------------------------------*/

#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include <conio.h>
#include <process.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <io.h>

void fatal();
char *malloc();
char *strcpy();
void exit();
int strlen();
int atoi();
void partition_image();
void scale_transformations();
void apply_transformations();
void smooth_image();
void read_transformations();
int outarray[100], countunpack;

unsigned char **image,*imptr,**image1;
double max_scale = 1.0;

int    s_bits, o_bits, min_part, max_part, dom_step,dom_step_type, dom_type;
int    only_positive, subclass_search, fullclass_search, *bits_needed;
int    zero_ialpha, output_partition, max_exponent;

struct transformation_node {
      int rx,ry,  xsize, ysize,  rrx,rry,  dx,dy;
      int sym_op;
      int depth;
      double scale, offset;
      struct transformation_node *next;
} transformations, *trans;

#define matrix_allocate(matrix, hsize,vsize, TYPE) {\
   TYPE *imptr; \
```

```
    int _i; \
    matrix = (TYPE **)malloc(vsize*sizeof(TYPE *));\
    imptr = (TYPE*)malloc((long)hsize*(long)vsize*sizeof(TYPE));\
    if (imptr == NULL) \
        fatal("\nNo memory in matrix allocate."); \
    for (_i = 0; _i<vsize; ++_i, imptr += hsize) \
        matrix[_i] = imptr; \
 }

#define bound(a)   ((a) < 0.0 ? 0 : ((a)>255.0? 255 : a))

FILE *input,*output,*other_input;

void fatal(s)
char *s;
{
    printf("\n%s\n",s);
    exit(-1);
}

long unpack(size, fin)
int size;
FILE *fin;
{
    int i;
    int value = 0;
    static int ptr = 1;
    static int sum;
    if (size == -2) {
        sum = fgetc(fin);
        sum <<= 1;
        return((long)0);
    }

    if (size == -1)
        return((long)((sum&256)>>8));

    for (i=0; i<size; ++i, ++ptr,  sum <<= 1) {
        if (sum & 256) value |= 1<<i;

         if (ptr == 8) {
            sum = getc(fin);
            ptr=0;
        }
    }
    return((long)value);
}
```

```
void main(argc,argv)
int argc;
char **argv;
{
   double        scalefactor;
   char          inputfilename[50];
   char          outputfilename[50];
   char          other_input_file[50];
   int           i,j, max_value, min_value, x_exponent, y_exponent;
   int           domain_size, no_domains;

   for (i=1; i<argc; ++i)
     if (argv[i][0] != '-' )
            if (inputfilename[0] == 1)
                  strcpy(inputfilename, argv[i]);
            else if (outputfilename[0] == 1)
                  strcpy(outputfilename, argv[i]);
            else;
     else {
            if (strlen(argv[i]) == 1) break;
            switch(argv[i][1]) {
              case 'i': strcpy(other_input_file,argv[++i]);
                        break;
              case 'n': num_iterations = atoi(argv[++i]);
                        break;
              case 'f': scalefactor = atof(argv[++i]);
                        break;
              case 'P': output_partition = 1;
                        break;
              case 'p': post_process = 0;
                        break;
              case 's': s_bits = atoi(argv[++i]);
                        break;
              case 'o': o_bits = atoi(argv[++i]);
                        break;
              case 'N': max_scale = atof(argv[++i]);
                        break;
              case '?':
              case 'H':
              default:
              fatal(" ");
            }
     }

   if ((input = fopen(inputfilename, "r")) == NULL)
         fatal("Can't open input file.");
```

```
scalefactor=1.0;
unpack(-2,input);

        countunpack=0;
min_part = (int)unpack(4,input);
        printf("min_part=%d \n", min_part);
max_part = (int)unpack(4,input);
        printf("miax_part=%d \n", max_part);
dom_step = (int)unpack(4,input);
        printf("dom_step=%d \n", dom_step);
dom_step_type = (int)unpack(1,input);
        printf("dom_step_type=%d \n", dom_step_type);
dom_type = (int)unpack(2,input);
        printf("dom_type=%d \n", dom_type);
hsize = (int)unpack(12,input);
        printf("hsize=%d \n", hsize);
vsize = (int)unpack(12,input);
        printf("vsize=%d \n", vsize);
max_value=(int)unpack(7, input);
min_value=(int)unpack(7, input);

x_exponent = (int)floor(log((double)hsize)/log(2.0));
y_exponent = (int)floor(log((double)vsize)/log(2.0));


max_exponent = (x_exponent > y_exponent ? y_exponent : x_exponent);
size =  1<<max_exponent;

zero_ialpha = 0.5 + (max_scale)/(2.0*max_scale)*(1<<s_bits);

scaledhsize = (int)(scalefactor*hsize);
scaledvsize = (int)(scalefactor*vsize);
matrix_allocate(image, scaledhsize,scaledvsize, unsigned char);
matrix_allocate(image1, scaledhsize, scaledvsize, unsigned char);

if (other_input_file[0] != 1) {
        other_input = fopen(other_input_file, "r");
        i = fread(image[0], sizeof(unsigned char),
                                scaledhsize*scaledvsize, other_input);
        printf("# bytes=%d\n", i);

        fclose(other_input);
}

 i = max_part - min_part + 1;
bits_needed = (int *)malloc(sizeof(int)*i);
no_h_domains = (int *)malloc(sizeof(int)*i);
```

```
domain_hstep = (int *)malloc(sizeof(int)*i);
domain_vstep = (int *)malloc(sizeof(int)*i);

   for (i=0; i <= max_part-min_part; ++i) {
   domain_size = size >> (min_part+i-1);
  if (dom_type == 2)
          domain_hstep[i] = dom_step;
  else if (dom_type == 1)
          if (dom_step_type ==1)
               domain_hstep[i] = (size >> (max_part - i-1))*dom_step;
          else
               domain_hstep[i] = (size >> (max_part - i-1))/dom_step;
  else
          if (dom_step_type ==1)
               domain_hstep[i] = domain_size*dom_step;
          else
               domain_hstep[i] = domain_size/dom_step;

  no_h_domains[i] = 1+(hsize-domain_size)/domain_hstep[i];

  if (dom_type == 2)
          domain_vstep[i] = dom_step;
  else if (dom_type == 1)
          if (dom_step_type ==1)
          domain_vstep[i] = (size >> (max_part - i-1))*dom_step;
          else
          domain_vstep[i] = (size >> (max_part - i-1))/dom_step;
  else
          if (dom_step_type ==1)
               domain_vstep[i] = domain_size*dom_step;
          else
               domain_vstep[i] = domain_size/dom_step;

  no_domains = 1+(vsize-domain_size)/domain_vstep[i];
  bits_needed[i] = ceil(log((double)no_domains*(double)no_h_domains[i])/
                        log(2.0));
 }

if ((output = fopen(outputfilename, "w")) == NULL)
          fatal("Can't open output file.");

 trans = &transformations;

fflush(stdout);
partition_image(0, 0, hsize,vsize );
fclose(input);
printf("Done.");
```

```
    fflush(stdout);

        if (scalefactor != 1.0) {
                printf("\nScaling image to %d x %d.", scaledhsize,scaledvsize);
                scale_transformations(scalefactor);
        }

    if (output_partition) {
          fprintf(output,"\n%d %d\n %d %d\n%d %d\n\n",
                    0, 0, scaledhsize, 0, scaledhsize, scaledvsize);
          printf("\nOutputed partition data in %s\n",outputfilename);
          fclose(output);
          return;
    }

    for (i=0; i<num_iterations; ++i)
      apply_transformations();

    if (post_process)
      smooth_image();

    i = fwrite(image[0], sizeof(unsigned char), scaledhsize*scaledvsize, output);
    if (i < scaledhsize*scaledvsize)
        fatal("Couldn't write output.");
    else
        printf("\n%d pixels written to output file.\n", i);

    fclose(output);
}

void read_transformations(atx,aty,xsize,ysize,depth)
int atx,aty,xsize,ysize,depth;
{
     int i,j, ialpha, ibeta;
    long domain_ref;

    double alpha, beta;

    if (depth < min_part) {
          read_transformations(atx,aty, xsize/2, ysize/2, depth+1);
          read_transformations(atx+xsize/2,aty, xsize/2, ysize/2, depth+1);
          read_transformations(atx,aty+ysize/2,xsize/2, ysize/2,  depth+1);
          read_transformations(atx+xsize/2,aty+ysize/2,xsize/2,ysize/2,depth+1);
          return;
    }

    if (depth < max_part && unpack(1,input)) {
```

```
        read_transformations(atx,aty, xsize/2, ysize/2, depth+1);
        read_transformations(atx+xsize/2,aty, xsize/2, ysize/2, depth+1);
        read_transformations(atx,aty+ysize/2, xsize/2, ysize/2, depth+1);
        read_transformations(atx+xsize/2,aty+ysize/2,xsize/2,ysize/2,depth+1);
} else {
       getchar();
        trans->next = (struct transformation_node *)
                  malloc(sizeof(struct transformation_node ));
        trans = trans->next; trans->next = NULL;
        ialpha = (int)unpack(s_bits,  input);
        ibeta = (int)unpack(o_bits,  input);

        alpha = (double)ialpha/(double)(1<<s_bits)*(2.0*max_scale)-max_scale;

        beta = (double)ibeta/(double)((1<<o_bits)-1)*
             ((1.0+fabs(alpha))*255);
        if (alpha > 0.0) beta -= alpha*255;

        trans->scale = alpha;
        trans->offset = beta;
        if (ialpha != zero_ialpha) {
          trans-> sym_op = (int)unpack(3, input);
          domain_ref = unpack(bits_needed[depth-min_part], input);
          trans->dx = (double)(domain_ref % no_h_domains[depth-min_part])
                                          * domain_hstep[depth-min_part];

          trans->dy = (double)(domain_ref / no_h_domains[depth-min_part])
                                          * domain_vstep[depth-min_part];
                printf("dy=%d ", trans->dy);
        } else {
          trans-> sym_op = 0;
          trans-> dx  = 0;
          trans-> dy = 0;
                printf("dx=%d ", trans->dx);
                printf("dy=%d ", trans->dy);
        }
        trans->rx = atx;
        trans->ry = aty;
        trans->depth = depth;
                printf("rx=%d ", trans->rx);
                printf("ry=%d ", trans->ry);
                printf("depth=%d ", depth);

        trans->rrx = atx + xsize;
        trans->rry = aty + ysize;
                printf("rrx=%d ", trans->rrx);
                printf("rry=%d ", trans->rry);
```

```
        if (output_partition)
                fprintf(output,"\n%d %d\n %d %d\n%d %d\n\n",
                  atx,      vsize-aty-ysize,
                  atx,      vsize-aty,
                  atx+xsize, vsize-aty);

    }
}

void apply_transformations()
{
    unsigned char **tempimage;
        int     i,j,i1,j1,count=0;
    double pixel;

    trans = &transformations;
    while (trans->next != NULL) {
        trans = trans->next;
                ++count;

#define COMPUTE_LOOP            {                               \
        pixel = (image[j1][i1]+image[j1][i1+1]+image[j1+1][i1]+     \
                image[j1+1][i1+1])/4.0;                         \
        image1[j][i] = bound(0.5 + trans->scale*pixel+trans->offset);\
        }

        switch(trans->sym_op) {
          case 0: for (i=trans->rx, i1 = trans->dx;
                        i<trans->rrx; ++i, i1 += 2)
                  for (j=trans->ry, j1 = trans->dy;
                        j<trans->rry; ++j, j1 += 2)
                    COMPUTE_LOOP
                  break;
          case 1: for (j=trans->ry, i1 = trans->dx;
                         j<trans->rry; ++j, i1 += 2)
                  for (i=trans->rrx-1,
                         j1 = trans->dy; i>=(int)trans->rx; --i, j1 += 2)
                    COMPUTE_LOOP
                  break;
          case 2: for (i=trans->rrx-1,
                        i1 = trans->dx; i>=(int)trans->rx; --i, i1 += 2)
                  for (j=trans->rry-1,
                         j1 = trans->dy; j>=(int)trans->ry; --j, j1 += 2)
                    COMPUTE_LOOP
                  break;
          case 3: for (j=trans->rry-1,
```

```
                        i1 = trans->dx; j>=(int)trans->ry; --j, i1 += 2)
                    for (i=trans->rx, j1 = trans->dy;
                        i<trans->rrx; ++i, j1 += 2)
                      COMPUTE_LOOP
                    break;
            case 4: for (j=trans->ry, i1 = trans->dx;
                        j<trans->rry; ++j, i1 += 2)
                    for (i=trans->rx, j1 = trans->dy;
                        i<trans->rrx; ++i, j1 += 2)
                      COMPUTE_LOOP
                    break;
            case 5: for (i=trans->rx, i1 = trans->dx;
                        i<trans->rrx; ++i, i1 += 2)
                    for (j=trans->rry-1,
                        j1 = trans->dy; j>=(int)trans->ry; --j, j1 += 2)
                      COMPUTE_LOOP
                    break;
            case 6: for (j=trans->rry-1,
                        i1 = trans->dx; j>=(int)trans->ry; --j, i1 += 2)
                    for (i=trans->rrx-1,
                        j1 = trans->dy; i>=(int)trans->rx; --i, j1 += 2)
                      COMPUTE_LOOP
                    break;
            case 7: for (i=trans->rrx-1,
                       i1 = trans->dx; i>=(int)trans->rx; --i, i1 += 2)
                    for (j=trans->ry, j1 = trans->dy;
                        j<trans->rry; ++j, j1 += 2)
                      COMPUTE_LOOP
                    break;
          }

    }
    tempimage = image;
    image = image1;
    image1 = tempimage;

    printf("\n%d transformations applied.",count);
}

void scale_transformations(scalefactor)
double scalefactor;
{
    trans = &transformations;
    while (trans->next != NULL) {
          trans = trans->next;

                trans->rrx *= scalefactor;
```

```
                trans->rry *= scalefactor;
                trans->rx *= scalefactor;
                trans->ry *= scalefactor;
                trans->dx *= scalefactor;
                trans->dy *= scalefactor;
    }
}

void partition_image(atx, aty, hsize,vsize )
int atx, aty, hsize,vsize;
{
  int x_exponent,
      y_exponent,
      exponent,
      size,
      depth;

  x_exponent = (int)floor(log((double)hsize)/log(2.0));
  y_exponent = (int)floor(log((double)vsize)/log(2.0));

  exponent = (x_exponent > y_exponent ? y_exponent : x_exponent);
  size = 1<<exponent;
  depth = max_exponent - exponent;

  read_transformations(atx,aty,size,size,depth);

  if (size != hsize)
    partition_image(atx+size, aty, hsize-size,vsize );

  if (size != vsize)
    partition_image(atx, aty+size, size,vsize-size );
}

void smooth_image()
{
   unsigned char pixel1, pixel2;
         int i,j; max_value1, min_value1, max_value2, min_value2)
         int w1,w2;

         printf("\nPostprocessing Image.");

   max_value1=0;
   min_value1=255;

   for(i=0; i<vsize; i++)
      for(j=0; j<hsize; j++) {
         if(image[j][i]<min_value1) min_value1=image[j][i];
```

```
        if(image[j][i]>man_value1) max_value1=image[j][i];
    }

    for(i=0; i<512; i++) {
                for(j=0; j<512; j++) {
                        image1[i][j]=(image[i][j]-max_value1)*(max_value-
                        min_value)/(max_value1-min_value1)+max_value+0.5;
                        if(image1[i][j]>max_value) image1[i][j]=max_ value;
                        if(image1[i][j]<min_ value) image1[i][j]=min_ value;
                        image[i][j]=image1[i][j];
                        }
        }
}
```

VITA

Ke Xiao was born in Beijing, China. When he was in eighth grade in middle school, like the whole generation of the Chinese people in 1960s and 1970s, his chance of accepting higher education was robbed by the 'Cultural Revolution' and he was sent to a factory as a 'blue collar' worker. After the 'reforming and opening' policy was carried out, he got the opportunity to take the entrance exams and was admitted by University of Science and Technology in Harbin, China. He graduated from this university in August 1982, with Bachelor of Engineering degree in Computer Science. After three years of working as a research associate in an institute in the Chinese Academy of Sciences, he returned to graduate school to accept more scientific training and graduate education. He completed his Master of Science degree in Mapping and Remote Sensing in Institute of Remote Sensing Application, Chinese Academy of Sciences, in August 1988. From 1988 to 1994, he was working in the Institute of Remote Sensing Application, Chinese Academy of Sciences as an assistant research professor and then an associate research professor. From October 1988 to December 1989, he was working as a visiting scientist in an international cooperation project "Geographic Information System of Remote Sensing and Mineral Exploration in Xinjiang Autonomous Region, North-West China" organized by The National Commission of Science and Technology of China and The Overseas Development Agency of Britain between China and Great Britain in Basingstoke, England. Since 1994, he has pursued a Doctor of Philosophy degree in Remote Sensing and Geographic Information System in the Department of Geography and Anthropology, Louisiana State University. He is currently the Information

Technology Project Coordinator in the Section of Medical Informatics, Louisiana State University Health Sciences Center, in New Orleans, Louisiana. The degree of Doctor of Philosophy will be conferred on him at the May 2003 Commencement.